DOI 10.37882/2223–2966.2023.03.38

БИБЛИОТЕКА ДЛЯ СОЗДАНИЯ ОТЧЕТОВ

Филяев Никита Алексеевич
Российский Технологический Университет МИРЭА
suchday1308@gmail.com
Корягин Сергей Викторович
Кандидат технических наук, «Московский государственный университет приборостроения и информатики»
dongenealog2003@mail.ru

LIBRARY FOR CREATING REPORTS

N. Filiaev
S. Koriagin

Summary. The existing options for the implementation of software tools in the framework of the development of a library for reporting are considered. A comparative analysis of existing software solutions is presented on the topic of the article. Based on the analysis of their advantages and disadvantages, requirements for the development of a software tool were formed and its target audience was determined. The library and the means for its use proposed by the authors are described. Using this library, which allows you to create report models in the required form, its interaction with external components is programmatically implemented. The library provides for the possibility of using ready-made algorithms to build individual reports. A set of algorithms built into the developed language is considered. The possibility of flexible expansion of functionality is provided. Examples of building user reports illustrating the functionality of the library proposed by the authors are considered.

Keywords: library for working with reports, broadcasting tool, graphical text processing.

Аннотация. Рассмотрены существующие варианты реализации программных средств в рамках разработки библиотеки для составления отчетов. По теме статьи представлен сравнительный анализ существующих программных решений. На основе анализа их достоинств и недостатков были сформированы требования к разработке программного средства и определена его целевая аудитория. Описаны предложенные авторами библиотека и средство для его использования. С использованием этой библиотеки, позволяющей в требуемом виде создать модели отчетов, программно реализуется ее взаимодействие внешними компонентами. Библиотека предусматривает возможность использования готовых алгоритмов, для построения индивидуальных отчетов. Рассмотрен набор встроенных в разработанный язык алгоритмов. Предусмотрена возможность гибкого расширения функционала. Рассмотрены примеры построения пользовательских отчетов, иллюстрирующих функциональные возможности предложенной авторами библиотеки.

Ключевые слова: библиотека для работы с отчетами, транслирующее средство, графическая обработка текста.

Введение

В настоящее время одной из актуальных задач является оперативная и автоматизированная работа с отчетностью, с помощью средств автоматизации. Наиболее актуальным и оптимальным средством для реализации подобных задач является язык C# и его модули.

Отчет — структурированная совокупность текстовых и графических объектов.

C# — высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности труда разработчика и читаемости кода. В то же время стандартная библиотека включает в себя большой объём полезных функций.

Библиотека в языке C#– это набор готовых функций, классов и объектов для решения каких-то задач.

Именно для этой цели появилось желание написать свою библиотеку, позволяющую пользователю по собственному желанию настраивать отчетность.

Язык C# достаточно богат функциями для работы с текстом, а также в нем присутствует множество библиотек с разным функционалом для работы с текстом, используя их как раз и появляется возможность разработать мощный инструмент для работы с отчетами.

Аналогов среди средств для многофункциональной работы с текстом, имеющих комбинированный набор всех необходимых функций, нет, либо они имею крайне ограниченное использование.

Современное решение проблемы

На данный момент для создания отчетности используются интегрированные в крупные ERM/CRM системы наподобие «Битрикс 24», «1С» сборщики отчетов,

```
<DocumentForm>
    <Info>
        <Name>название</Name>
        <ShortName>Короткое название</ShortName>
        <Description>Описание</Description>
        <Uid>9147a569-28aa-40e5-b453-32a3adcb4633</Uid>
    </Info>
    <Params>
        <ColumnCount>12</ColumnCount>
        <RowsCount>25</RowsCount>
        <Orientation>horizontal</Orientation>
    </Params>
    <Structure>
```

Рис. 1. Основные параметры отчета

```
    <Block X="0" Y="3" Width="12" Height="2" ContentType="text">
        <Style>
            <FontSize>2</FontSize>
            <FontName>Arial</FontName>
            <Underline>false</Underline>
            <Bold>false</Bold>
            <Italic>false</Italic>
            <Strikeout>false</Strikeout>
            <ForeColor>#000000</ForeColor>
            <BackColor>#FFFFFF</BackColor>
            <TextAlign>Center</TextAlign>
        </Style>
        <Text>Какой-то текст</Text>
```

Рис. 2. Стиль блока и его содержимое

```
    <Structure>
        <Column Title="Заголовок 1" Name="1 column" ColSpan = "1"/>
        <Column Title="Заголовок 2" Name="2 column" ColSpan = "1"/>
        <Column Title="Заголовок 3" Name="3 column" ColSpan = "1"/>
        <Column Title="Заголовок 4" Name="4 column" ColSpan = "1"/>
        <Column Title="test поле" Name="test" ColSpan = "1"/>
    </Structure>
    <Data>
        <Row>
            <2 column>1</2 column>
            <1 column>1</1 column>
            <3 column>rbebr</3 column>
            <4 column></4 column>
<test></test>
        </Row>
        <Row>
            <2 column>2</2 column>
            <1 column>2</1 column>
            <3 column>BMreberbreФ</3 column>
            <4 column></4 column>
<test></test>
        </Row>
        <Row>
            <2 column>3</2 column>
            <1 column>3</1 column>
            <3 column>BnernrenBC</3 column>
            <4 column></4 column>
```

Рис. 3. Пример структуры блока таблицы

```
Ссылок: 1
public void GetReport(System.Xml.XmlDocument _document,
                      string exportFormat,
                      string savePath)
{
    _savePath = savePath;
    if (_document != null)
    {
        var document = _document;
        XDocument xdoc = _document.ToXDocument(); // Данные для создания моделей блоков
        XmlElement xRoot = document.DocumentElement; // Данные для формирования содержания блоков
        CreateReport(xRoot, xdoc, exportFormat);
    }
    else
    {
        Debug.WriteLine("файл отсутствует или является пустым");
    }
}
```

Рис. 4. Код метода GetReport

```
Ссылок: 0
public void GetReportToStream(System.Xml.XmlDocument _document, string exportFormat,
                      MemoryStream reportStream)
{
    //_savePath = savePath;
    if (_document != null)
    {
        var document = _document;
        XDocument xdoc = _document.ToXDocument(); // Данные для создания моделей блоков
        XmlElement xRoot = document.DocumentElement; // Данные для формирования содержания блоков
        CreateReport(xRoot, xdoc, exportFormat, reportStream);
    }
    else
    {
        Debug.WriteLine("файл отсутствует или является пустым");
    }
}
```

Рис. 5. Код метода GetReportToStream

```
Ссылок: 2
72  public void CreateReport(XmlElement xRoot,
73                           XDocument xdoc,
74                           string setExport,
75                           MemoryStream reportStream = null)
76  {
77      using (Report testReport = new Report())
78      {
79          //testReport.ScriptText = "using System.Data.dll";
80
81          Border border = new Border
82          {
83              Lines = BorderLines.All,
84              Width = 3
85          };
86
87          Page
102
103         Data
112
113         try
114         {
115             var node = GetModel(xdoc, xRoot);
116
117             foreach (var model in node) // model блок который надо построить
118             {
```

Рис. 6. Код метода CreateReport

```
private IEnumerable<BlockModel> GetModel(XDocument xdoc, XmlElement xRoot)
{
    // Количество столбцов на странице отчета
    float pageColumnsCount = float.Parse(xRoot.SelectSingleNode("Params//ColumnCount").InnerText);
    // Количество строк на странице отчета
    float pageRowsCount = float.Parse(xRoot.SelectSingleNode("Params//RowsCount").InnerText);

    string fontFamily = xRoot.SelectSingleNode("Params//FontFamily").InnerText;

    /// Удалить после добавления в xml картинки ...
    ////////////////////////////////////

    // в node храниться разметка и содержание блока
    var node = from val in xdoc.Element("DocumentForm").Element("Structure").Elements("Block")
               select new BlockModel
               {
                   // Разметка
                   type = val.Attribute("ContentType").Value,
                   PointX = val.Attribute("X").Value,
                   PointY = val.Attribute("Y").Value,
                   _Width = val.Attribute("Width").Value,
                   _Height = val.Attribute("Height").Value,
                   // Содержание
                   _Text = val.Element("Text")?.Value,
                   _Value = val.Element("Value")?.Value,
                   _pageColumnsCount = pageColumnsCount,
                   _pageRowsCount = pageRowsCount,
                   _srcImage = val.Element("Image")?.Value,
                   // Стиль
                   _fontName = fontFamily, //val?.Element("Style")?.Element("FontName")?.Value,
                   _fontSize = val?.Element("Style")?.Element("FontSize")?.Value,
                   _underline = val?.Element("Style")?.Element("Underline")?.Value,
                   _bold = val?.Element("Style")?.Element("Bold")?.Value,
                   _italic = val?.Element("Style")?.Element("Italic")?.Value,
                   _strikeout = val?.Element("Style")?.Element("Strikeout")?.Value,
                   _foreColor = val?.Element("Style")?.Element("ForeColor")?.Value,
                   _backColor = val?.Element("Style")?.Element("BackColor")?.Value,
                   _textAlign = val?.Element("Style")?.Element("TextAlign")?.Value,
                   modelStruct = val,
               };
```

Рис. 7. Код метода GetModel

```
namespace Start
{
    Ссылок: 0
    class Program
    {
        Ссылок: 0
        static void Main(string[] args)
        {
            DocumentGenerator.DocumentGenerator gen = new DocumentGenerator.DocumentGenerator();

            var document = new XmlDocument();
            string path = Path.Combine(Environment.CurrentDirectory);
            document.Load(path + "/TestDocForm.xml");

            MemoryStream stream = new MemoryStream();

            gen.GetReport(document, "pdf", @"E:\path\to\save\");
            //gen.GetReportToStream(document, stream);

            Console.WriteLine(stream.Length);
        }
    }
}
```

Рис. 8. Пример использования методов библиотеки в сторонней программе

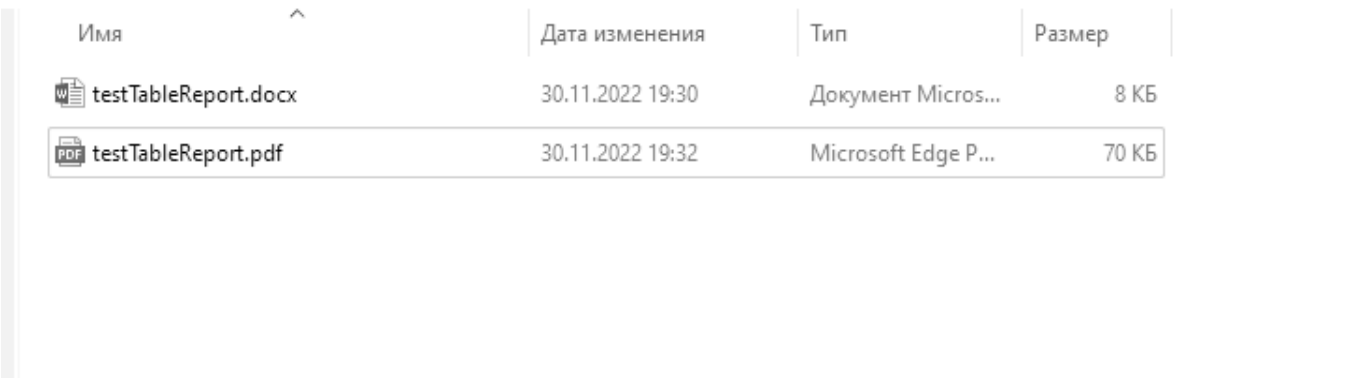

Рис. 9. Файлы отчетов

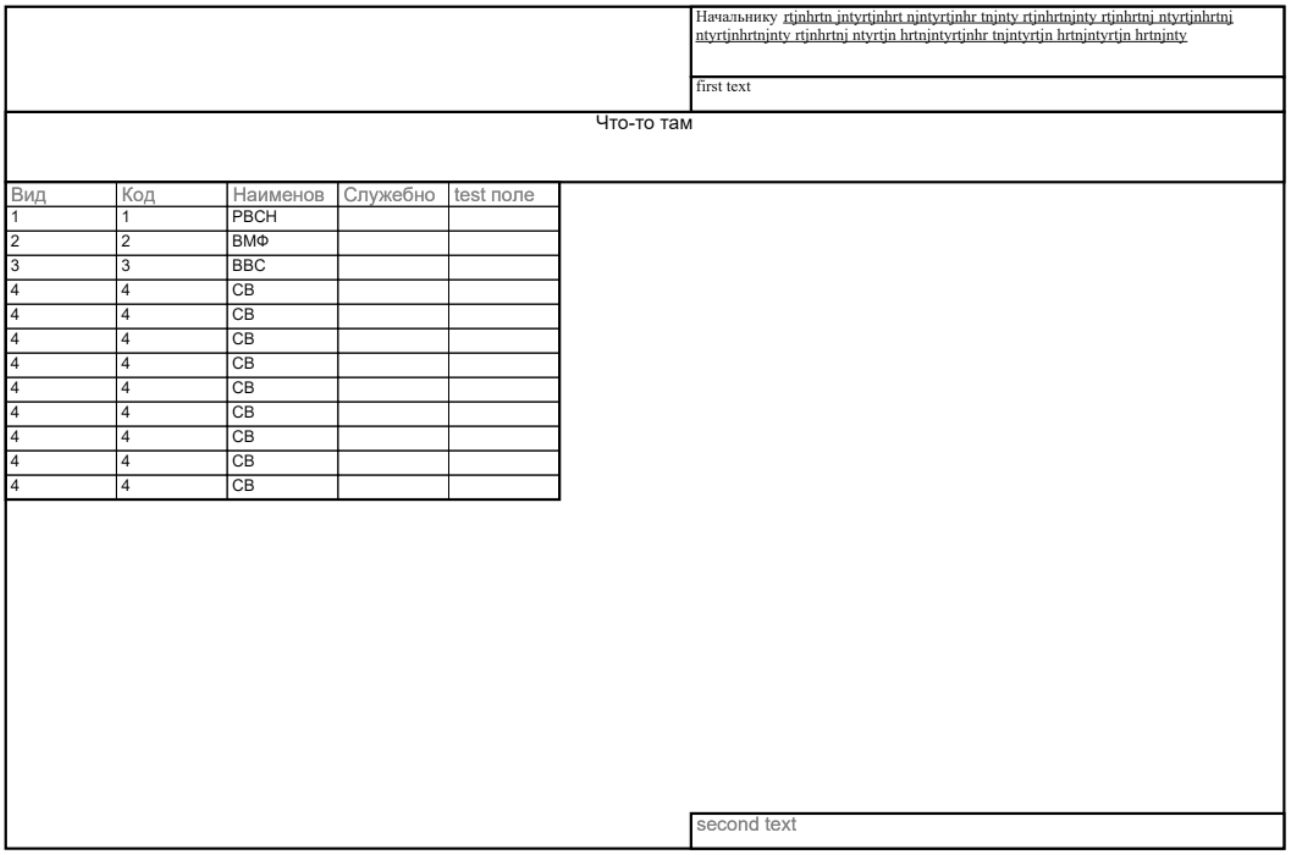

Рис. 10. Отчет в формате pdf с разметкой блоков

но для их использования необходимо внедрять данные системы. Также данные сборщики ограничены строгими рамками, поскольку в них структура отчетов заложена заранее.

В связи с описанными выше причинами было решено разработать новое программное средство. Оно должно быть достаточно простым, чтобы им мог воспользоваться любой пользователь, а также иметь возможность расширять свой функционал, к примеру, для дополнительной графической обработки текста.

Для этого предлагается разработать свою библиотеку и соответствующее ему транслирующее средство, с помощью которого, в свою очередь, будет обработана текстовая информация. В случае необходимости име-

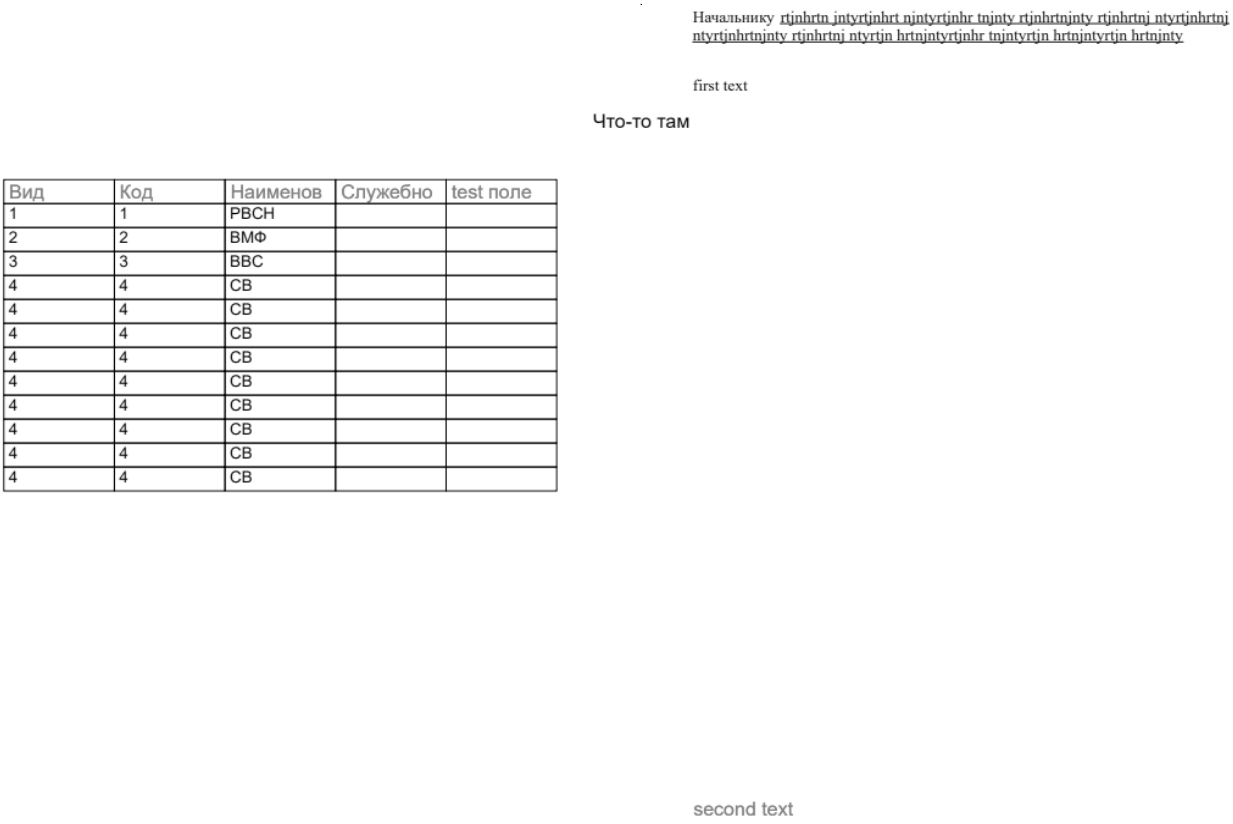

Рис. 11. Отчет в формате pdf

ется возможность применить предлагаемый подход и языковое средство для иных, аналогичных приложений, с доработкой функциональности в зависимости от требований к конечному программному продукту.

Описание структуры входных данных

Формальное описание входных данных, представлено ниже:

Модель XML строки, подающейся на вход:

Предлагаемая структура позволяет буквально собирать отчет по кусочкам, каждый из которых можно настроить под свои нужды.

Каждый блок содержит:

- «ContentType» — тип объекта (текст, таблица и т.д.);
- «Style» — параметры текста внутри объекта;
- «Structure» — структура объекта;
- «Data» — данные для объекта, параметр используется в основном для составления таблиц.

Также в начале XML есть основные параметры отчета:

- «Info» — краткое описание отчета;
- «Params» — основные параметры отчета.

Основные функции и методы библиотеки

«GetReport» — получение отчета в формате, определенном пользователем.

«GetReportToStream» — получение объекта отчета в виде массива байтов.

«CreateReport» — создание структуры и объектов отчета, также объединение их в конечный файл заданного формата.

«GetModel» — десериализация XML объекта в удобный для обработки вид.

Примеры работы программного обеспечения

Рассмотрим пример с использованием разработанного программного обеспечения. На рис. 8 представлен пример работы приложения.

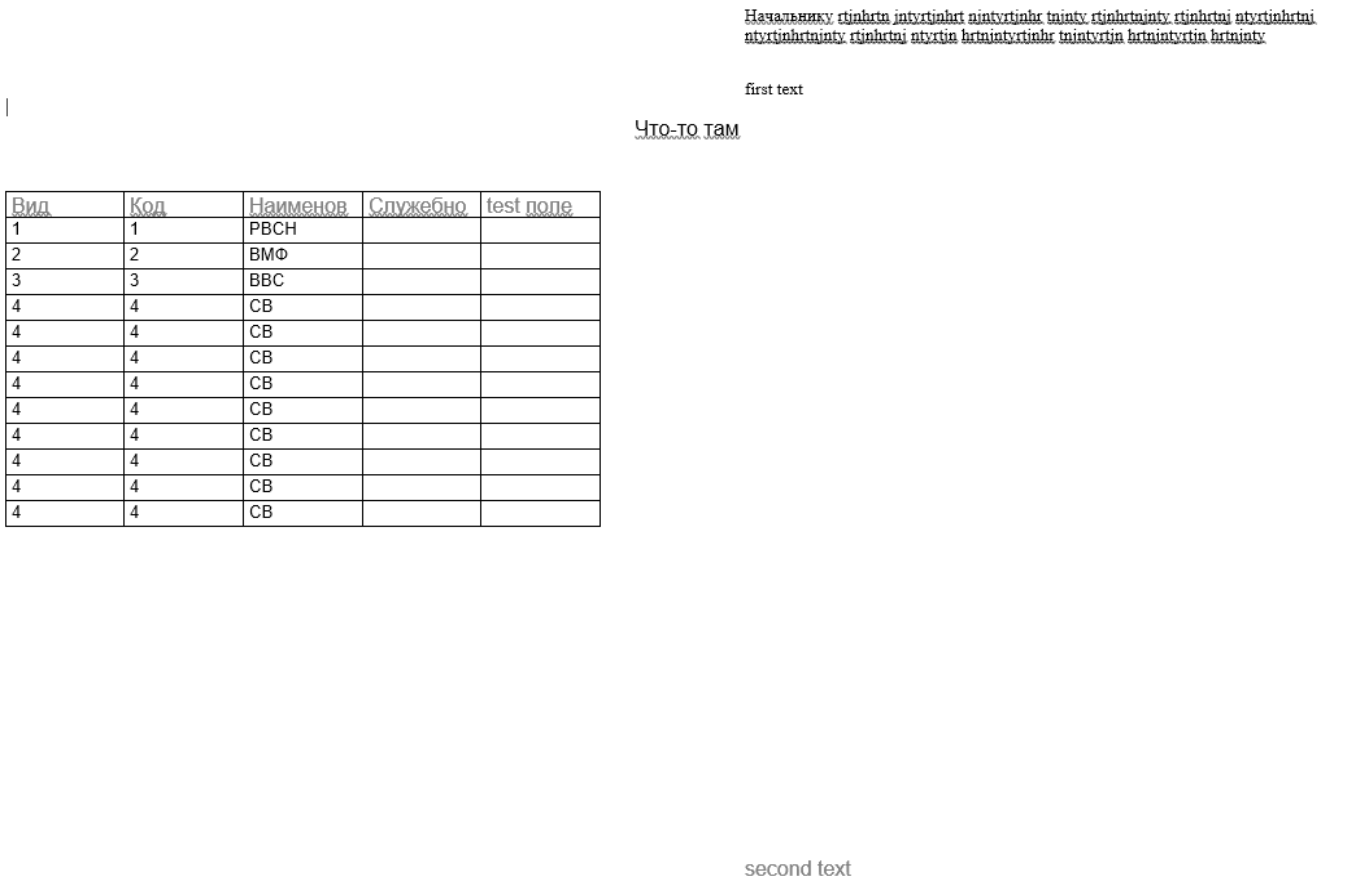

Рис. 12. Отчет в формате docx

После выполнения программы отчеты будут лежать в папке, которую указал пользователь (рис. 8—12).

Выводы

Выше были рассмотрены примеры с использованием библиотеки, в каждом из них программное обеспечение отработало корректно.

В результате проведенной работы была разработана библиотека как интеграционное решение для системы, которая сможет подавать на вход данные в нужном формате. В данное средство заложен необходимый функционал, требуемый для формирования результатов, а также присутствует возможность и дальше расширять функционал представленного программного продукта.

В реализованном приложении используются принципы объектно-ориентированного программирования, что предоставит необходимую гибкость и корректность в дальнейшей обработке отчетов.

В качестве расширения планируется сделать программу самостоятельной, то есть добавить графический интерфейс и интерфейс взаимодействия.

ЛИТЕРАТУРА

1. Свердлов С.З. Языки программирования и методы трансляции, уч. пособ./ Свердлов С.З. — СПб: Питер, 2007. — 638 с.
2. https://www.fastreport.com/public_download/docs/FRNet/FRNetProgrammerManual
3. Рихтер Дж CLR via C#. Программирование на платформе Microsoft.NET Framework 4.0 на языке C#. 3-е изд. — СПб.: Питер, 201 2. — 928 с.: ил.
4. Скиена С. Алгоритмы. Руководство по разработке. — 2-е изд.: Пер. с англ. — СПб.: БХВ-Петербург, 2011. — 720 с.: ил.

Журнал «Современная наука: актуальные проблемы теории и практики»