УДК 004.624, 004.912

ИМПОРТ И ЭКСПОРТ ДАННЫХ ИЗ ПРИЛОЖЕНИЙ, НАПИСАННЫХ НА ЯЗЫКЕ ПРОГРАММИРОВАНИЯ C#, В ДОКУМЕНТЫ MS WORD И MS EXCEL

А.В. ВОЙТЕХОВИЧ
(Представлено: канд. физ.-мат. наук, доц. О.В. ГОЛУБЕВА)

Исследуется вопрос оптимизации ручного труда при работе с данными, хранящимися в электронном виде. Также автор рассматривает методы решения этой проблемы применительно к языку программирования C# и документам MS Word и MS Excel.

Сегодня большая часть документации, которая передается между людьми, отделами и учреждениями, хранится в электронном формате. Это позволяет сократить расходы на печать документов, легко вносить в них поправки, создавать новые документы на основе шаблонных и собирать информацию из многих источников в единый ресурс. При этом не всегда всю информацию удобно хранить только в текстовом или только в табличном формате. При работе с документами периодически возникают проблемы дублирования данных, различных значений для одного и того же понятия в разных копиях документов (например, указаны разные даты для одного события или различная стоимость в двух накладных по одному товару), проблемы некорректного отображения данных и тому подобные. К тому же при работе с числовыми данными в текстовых документах часто приходится производить вычисления вручную, а при обнаружении ошибки повторять весь процесс заново.

Для решения проблем обработки большого объема данных разработано бессчетное множество программ, которые обеспечивают ввод, хранение, обработку и вывод данных, связанных с указанной предметной областью. Все они работают с прикрепленными к ним базами данных, которые обеспечивают корректное хранение данных без излишнего их дублирования. Такие системы можно разделить на два типа: справочные системы; автоматизированные информационные системы [1].

Справочные системы используются для получения справочной информации. Администратор или ответственное лицо вносят данные в систему вручную, а пользователи с помощью модуля поиска находят и просматривают нужную им в данный момент информации. Примером таких систем являются справочные терминалы на железнодорожных и автобусных вокзалах, электронные каталоги, справочные терминалы в книжных магазинах.

Автоматизированные информационные системы, в отличие от справочных, могут не только обрабатывать и хранить информацию, но и получать в качестве входных данных документы, оформленные в установленном формате, а также генерировать отчеты, выгружать их в документы различных форматов и печатать. Некоторые информационные системы кроме этих функций могут выполнять и иные, более специфические для своих предметных областей, другие же поддерживают только часть из вышеперечисленных функций.

Для того чтобы обеспечить передачу данных между автоматизированными информационными системами и внешними документами, необходимо изучить принципы импорта и экспорта данных в различных языках программирования. В случае импорта или экспорта в файлы формата txt, xml и тому подобных достаточно функций чтения и записи в файл, открытый в текстовом режиме. Разница состоит только в представлении данных – в текстовом файле данные будут структурированы с помощью пробелов, символов переноса строки или специальных, заранее условленных, символов и их последовательностей, а в xml-подобных файлах – с помощью тегов.

В случае работы с doc- и xls-файлами дела обстоят хуже, так как эти файлы помимо полезных данных хранят большой объем информации о форматировании текста и разметке документов. Для того чтобы стороннее приложение получило доступ к данным в таком формате, лучше всего использовать специализированные библиотеки. Библиотеки для работы с файлами MS Word и MS Excel для разных языков программирования работают по единому принципу. Файл открывается в фоновом режиме для чтения или записи соответствующим приложением (Word или Excel) и передача данных производится не напрямую между документом и пользовательским приложением, а по цепочке: пользовательское приложение -> функции из специализированной библиотеки -> приложение для работы с документами данного типа -> документ и обратно.

Рассмотрим функции специализированных библиотек импорта и экспорта данных для приложений, написанных на языке программирования C# Microsoft.Office.Interop.Excel и Microsoft.Office.Interop.Word. Стоит отметить, что эти библиотеки не входят в стандартный пакет библиотек и их необходимо скачать и подключить к проекту вручную. При этом нужно учесть некоторые их особенности. В связи с тем, что сейчас активно используются минимум три версии пакета Microsoft Office (2007, 2010, 2013), а в некоторых учреждениях до сих пор используется пакет 2003 года, нужно скачать библиотеку, предназначенную

для конкретной версии пакета. Это связано с тем, что в новых версиях постоянно добавлялись новые макросы, элементы разметки документа, а также постепенно менялся формат представления данных. Тем не менее часть функций осталась неизменной. На основе этого была организована обратная совместимость документов старых версий с новыми пакетами. Таким образом, библиотека, предназначенная для пакета 2010, будет корректно работать на компьютере, на котором установлен MS Office 2010, а если будет установлен пакет более ранней версий, она может некорректно отображать часть информации и выбрасывать ошибки в случае вызова функций, отсутствующих в приложении. Если же на компьютере будет установлен MS Office 2013, то приложение не сможет связаться с документом, выбрасывает ошибку и завершает свою работу.

Работа с текстовыми и табличными файлами проводится аналогично, поэтому рассмотрим импорт и экспорт данных на примере работы с документом MS Excel [2].

После скачивания библиотеки ее необходимо переместить в папку ресурсов и подключить к проекту. Программист получает возможность использовать все функции из подключенных библиотек, но их вызов сопровождается префиксом Microsoft.Office.Interop.Excel или Microsoft.Office.Interop.Word. Для того чтобы укоротить запись функций и получить возможность подключить другую версию специализированных библиотек, не меняя исходного кода всего приложения, перед описанием главного класса приложения можно прописать глобальный алиас:

using Exc = Microsoft.Office.Interop.Excel.

Алиас – это специфический синоним названию библиотеки, прописанный программистом вручную, который используется на равных правах со стандартным названием библиотеки. После добавления вышеописанного алиаса функции работы с документами MS Excel можно вызывать с коротким префиксом – Exc.

Приложение MS Excel, с которым работает пользовательское приложение, работает на основе древовидной структуры, корень которой – объект Application. Этот объект содержит одну или более книг, ссылки на которые хранятся в свойстве Workbooks. Каждая книга является объектом Workbook, которая, в свою очередь, содержит одну или более страниц, ссылки на которые хранятся в свойстве Worksheets, и может содержать диаграммы – свойство Charts. Страницы содержат объекты ячейки или группы ячеек, ссылки на которые доступны через объект Range. Еще ниже в иерархии строки и столбцы. Объект Chart содержит ссылки на серии линий, легенды и тому подобные элементы.

В первую очередь, при работе с документами необходимо создать глобальный объект класса Application, затем программист получает возможность открыть его в режиме чтения или записи либо создать новый документ.

Ниже представлен фрагмент программы, которая открывает существующий файл, объединяет ячкейки A1–E1, добавляет в них надпись «Тестовый запуск прошел успешно», сохраняет изменения и закрывает файл. Каждая строка исходного кода пронумерована для удобства объяснения действий, производимых в той или иной строке.

```
1:  Excel.Application excel = new Excel.Application();
2:  excel.Visible = false;
3:
4:  Excel.Workbook fileExcel;
5:  fileExcel = excel.Workbooks.Open("C:/file1.xlsx");
6:
7:  Excel.Sheets excelsheets =  excelapp.Worksheets;
8:  Excel.Worksheet excelworksheet = (Excel.Worksheet)excelsheets.get_Item(1);
9:
10: Excel.Range excelcells=excelworksheet.get_Range("A1",Type.Missing);
11: excelcells=excelcells.get_Offset(0,4);
12: excelcells.Value2 = “Тестовый запуск прошел успешно”;
13:
14: fileExcel.Save();
15: fileExcel.Close();
16: excel.Quit();
```

В первой строке инициализируется глобальный объект, через который программа получит доступ к документу Excel и одновременно запускается приложение MS Excel в фоновом режиме. Во второй строке мы указываем, что фоновое окно, в котором будет открыт файл, не будет отображаться на экране. Если вместо «false» указать «true», то окно будет выведено на экран без связанного с ним документа. В строках 4–5 происходит создание и инициализация переменной, хоторая хранит ссылку на рабочую книгу. Функция excel.Workbooks.Open открывает документ, полный путь к которому указан в круглых скобках. Если файл поврежден, отсутствует или имеет неверное расширение, то приложение выдаст ошибку и автоматически закроется. Такие ошибки необходимо отслеживать, но в рамках простейшего примера в этом необходимости нет.

В переменной excelsheets (строка 7) будут храниться ссылки на все листы, существующие в ранее открытой книге, а в переменную excelworksheet в строке 8 заносится ссылка на первый лист книги с помощью функции get_Item(). Если в качестве параметра указать не единицу, а другое число, будет открыта страница с указанным порядковым номером. При этом стоит предусмотреть проверку на корректность указанного числа, так как оно не может быть нулевым, отрицательным или большим, чем количество листов в открытой книге.

Функция get_Range() возвращает ссылку на ячейку, указанную в первом параметре (строка 10), а функция get_Offset() позволяет выделить диапазон ячеек и объединить их. Диапазон указывается с помощью двух чисел: первое показывает, сдвигается ли граница диапазона вниз относительно нижнего края ячейки и насколько, а второе – сдвигается ли граница диапазона вправо относительно правого края ячейки и насколько. В случае указания отрицательных чисел сдвиг происходит в обратную сторону. В строке 11 представленного исходного кода в переменную excelcells вместо ссылки на ячейку A1 заносится ссылка на новообразованный диапазон, который состоит из объединенных ячеек A1, B1, C1, D1 и E1. Значение ячейки устанавливается с помощью свойства Value2, как показано в строке 12. Далее остается только сохранить внесенные изменения, закрыть документ и разорвать связь с приложением MS Excel с помощью функции Quit(). Данный процесс описан в строках 14–16 исходного кода. Считывание данных происходит аналогичным образом. Результат выполнения данной программы представлен на рисунке 1.

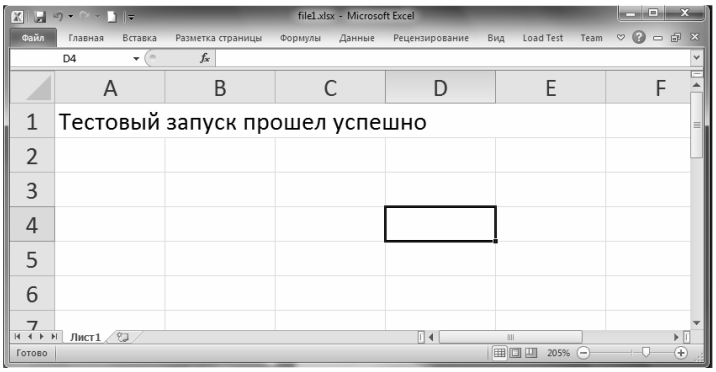

Рисунок 1. – Результат работы программы

При работе с текстовыми документами происходит тот же процесс. Разница состоит лишь в иерархии объектов, вложенных в корневой объект Applications. Эта иерархия представлена на рисунке 2.

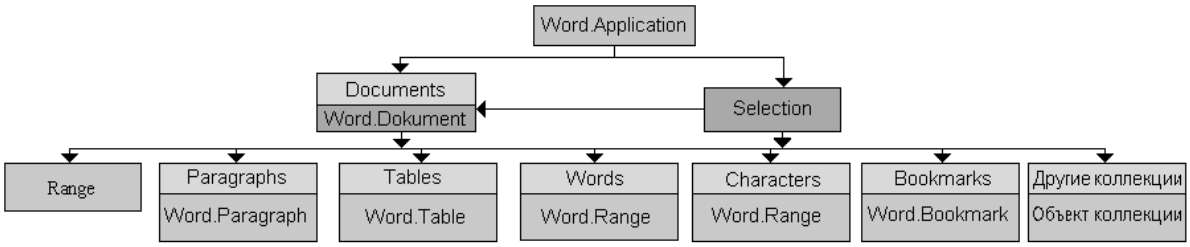

Рисунок 2. – Иерархия основных объектов Word, необходимых при разработке приложения

Таким образом, можно сказать, что изучение импорта и экспорта данных из приложения, написанного на языке программирования C#, в текстовые и табличные документы сводится к изучению иерархии объектов в корневом элементе Applications для Word и Excel, а также изучению основных принципов и функций работы с ними. Несмотря на то, что при первом рассмотрении тема кажется слишком обширной и трудной, после изучения базового материала и нескольких попыток применить полученные знания на практике выясняется, что работа с внешними документами сводится к механическому переписыванию шаблонных наборов функций, а самая большая проблема при работе с ними – это опасность запутаться в очередности и позиции выводимых и считываемых данных.

ЛИТЕРАТУРА

1. Википедия – свободная энциклопедия: Информационная система [Электронный ресурс]. – Режим доступа: https://ru.wikipedia.org/wiki/Информационная_система. – Дата доступа: 10.04.2016.
2. Работа с серверами автоматизации Word и Excel в Visual Studio .Net [Электронный ресурс]. – Режим доступа: http://wladm.narod.ru/C_Sharp/componentbegin.html. – Дата оступа: 10.06.2016.