# **Практическая работа № 11.** Использование функций MSWord, MSExcel. Редактирование таблицы БД MSAccess в среде VisualStudio

Объём учебного времени – 2ч

Методические рекомендации

1. Цель работы: Научиться использовать функции MSWord, MSExcel в программах на C#.
2. Перечень необходимых средств обучения:
   1. Технические средства обучения:

Компьютер Core i3 3.0, 2 Gb оперативной памяти, винчестер 250 Gb, DVD

* 1. Программное обеспечение:
* ОС Windows 10
* MicrosoftOffice
* MS Visual Studio 2017

1. Основные теоретические положения:

Пакет приложений MicrosoftOffice является сервером OLE-объектов, его функциями могут пользоваться другие приложения.

**Работа с MS Word из C#**

Существует два класса - WordDocument и WordSelection (просто часть документа, обертка над Range), реализующие приемы работы с текстом, тестовый шаблон и окно, позволяющее протестировать работу классов и наглядно увидеть, как именно все цепляется к событиям формы.

**Работа с** таблицами. Выбрать существующую таблицу внутри документа можно через существующий интерфейс Tables по её порядковому номеру. Приэтоммыполучаемобъекттипа Table.

**Word.Table \_table = \_document.Tables[tableNumber];**

Новая таблица вставляется методом **Tables.Add.** Строки добавляются методом table.Rows.Add().Тип границ для таблицы можно задать так:

**\_table.Borders.OutsideLineStyle = Word.WdLineStyle.wdLineStyleDouble;**

**\_table.Borders.InsideLineStyle = Word.WdLineStyle.wdLineStyleDouble;**

Имеятаблицумыможемполучитьдиапазондляконкретнойячейкипономерустроки/колонкичерезинтерфейс Cell: **\_currentRange = \_table.Cell(rowIndex, columnIndex).Range;**

**Работа с MS Excel из C#**

Табличный процессор MSExcel содержит мощные средства для сложных вычислений и анализа данных, которые могут расширить возможности программ. **MicrosoftExcel**— программа для работы с электронными таблицами, созданная корпорацией **Microsoft** для **MicrosoftWindows**. Она предоставляет возможности экономико-статистических расчетов, графические инструменты и язык макропрограммирования **VBA** (**VisualBasic** для приложений). **MicrosoftExcel** входит в состав **MicrosoftOffice** и на сегодняшний день **Excel** является одним из наиболее популярных приложений в мире.

Для автоматизации работы с **Office** клиентская программа либо создает новый экземпляр приложения, либо получает ссылку на уже существующий. Как правило, корпорация Microsoft рекомендует использовать первый вариант. Однако в определенных ситуациях возникает необходимость автоматизации уже запущенного экземпляра приложения Office. В этом случае клиент автоматизации получает из таблицы **ROT** (RunningObjectTable) ссылку на **COM**-объект сервера автоматизации. После получения ссылки или же запуска становится доступна вся объектная модель **Excel**. С точки зрения программиста она выглядит так:

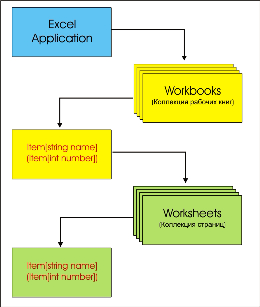


Рисунок 1 Объектная модель MSExcel

Для работы вместе с объектом **Excel** необходимо получить ссылку на его коллекцию книг. С ее помощью можно получить доступ к любой книге. У каждой книги есть коллекция страниц, ссылку на которую также нужно получить для доступа к конкретной странице. Доступ к книгам и к страницам можно получить как по их имени, так и по их порядковому номеру.

Пример 1. Импорт данных из MSExcel

**//Диалоговое окно выбора файла с фильтром**

**OpenFileDialogopenDialog = new OpenFileDialog();**

**openDialog.Filter = "Файл Excel|\*.XLSX;\*.XLS";**

**openDialog.ShowDialog();**

**try**

**{**

**//Приложениесамого Excel**

**ObjExcel = new Microsoft.Office.Interop.Excel.Application();**

**//Книга.**

**ObjWorkBook = ObjExcel.Workbooks.Open(openDialog.FileName);**

**//Таблица.**

**ObjWorkSheet = ObjExcel.ActiveSheet as Microsoft.Office.Interop.Excel.Worksheet;**

**//Ячейка**

**Microsoft.Office.Interop.Excel.Rangerg = null;**

**Int32 row = 1;**

**dataGridViewMain.Rows.Clear();**

**while (ObjWorkSheet.get\_Range("a" + row, "a" + row).Value != null)**

**{**

**// Читаемданныеизячейки**

**rg = ObjWorkSheet.get\_Range("a" + row, "c" + row);**

**foreach (Microsoft.Office.Interop.Excel.Range item in rg)**

**{**

**try**

**{**

**arr.Add(item.Value.ToString().Trim());**

**}**

**catch { arr.Add(""); }**

**}**

**dataGridViewMain.Rows.Add(arr[0], arr[1], arr[2]);**

**arr.Clear();**

**row++;**

**}**

**MessageBox.Show("Файлуспешносчитан!", "Считыванияexcelфайла");**

**}**

**catch (Exception ex) { MessageBox.Show("Ошибка: " + ex.Message, "Ошибкаприсчитывании excel файла"); }**

Пример 2. Экспорт данных в Excel:

**fileName = System.Windows.Forms.Application.StartupPath + "\\" + textBoxFileName.Text + ".xlsx";**

**try**

**{**

**//Приложениесамого Excel**

**ObjExcel = new Microsoft.Office.Interop.Excel.Application();**

**//Книга.**

**ObjWorkBook = ObjExcel.Workbooks.Add(System.Reflection.Missing.Value);**

**//Таблица.**

**ObjWorkSheet = (Microsoft.Office.Interop.Excel.Worksheet)ObjWorkBook.Sheets[1];**

**for (int i = 0; i<dataGridViewMain.Rows.Count; i++)**

**{**

**DataGridViewRow row = dataGridViewMain.Rows[i]; // строки**

**for (int j = 0; j <row.Cells.Count; j++) //циклпоячейкамстроки**

**{**

**ObjExcel.Cells[i + 1, j + 1] = row.Cells[j].Value;**

**}**

**}**

**ObjWorkBook.SaveAs(fileName);**

**}**

**catch (Exception ex) { MessageBox.Show(ex.Message, "Error");}**

1. Практические указания:
   1. Проверка правописания в текстовом поле с помощью обращения к MSWord
      1. В меню Средства выберите **Диспетчер пакетов NuGet – Консоль диспетчера пакетов**. Для добавления пакета Intereop.Wordв проектвведите: 
      2. Добавьте ссылку в проект (см. рисунок ниже).

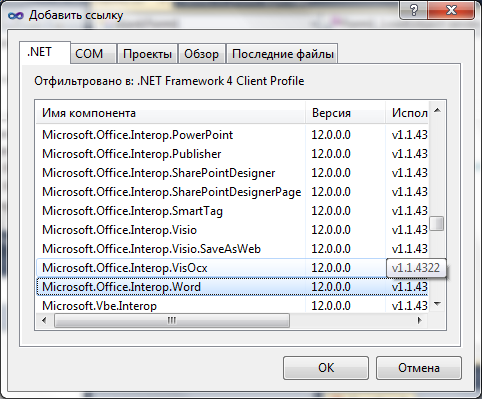


Рисунок 2 Окно добавления ссылки в проект

* + 1. Добавьте в проект окно ввода **textBox1**, установите для него возможность многострочного ввода данных, добавьте на форму кнопку, переименуйте её на Проверка орфографии.
    2. На событие загрузки формы напишите процедуру:
       - Очистите окно ввода;
       - Свойству **TabIndex** компонента **textBox1** присвойте значение 0;
       - Свойству **TabIndex** компонента **button1** присвойте значение 1;
    3. Напишите процедуру для нажатия на кнопку Проверка орфографии:
       - Опишите новую переменную **Word1** как new**Microsoft.Office.Interop.Word.Application**();
       - Опишите переменную **Doc1** равную **Word1**.**Documents**.**Add**();
       - С помощью метода **InsertBefore** для объекта **Doc1.Words.First** добавьте текст из окна ввода;
       - С помощью метода **CheckSpelling**() для объекта **Doc1** проверьте орфографию в документе;
       - Опишите новую переменную **isp**\_**text**, инициализируйте её как **Doc1.Content.Text**;
       - Отобразите в окне ввода исправленный текст;
       - Закройте документ MSWord, не сохраняя изменений: **Word1.Documents.Close(false);**
       - Напишите инструкцию для выхода из приложения MSWord;
       - Переменной Word1 присвойте пустое значение;
    4. В результате мы получим:

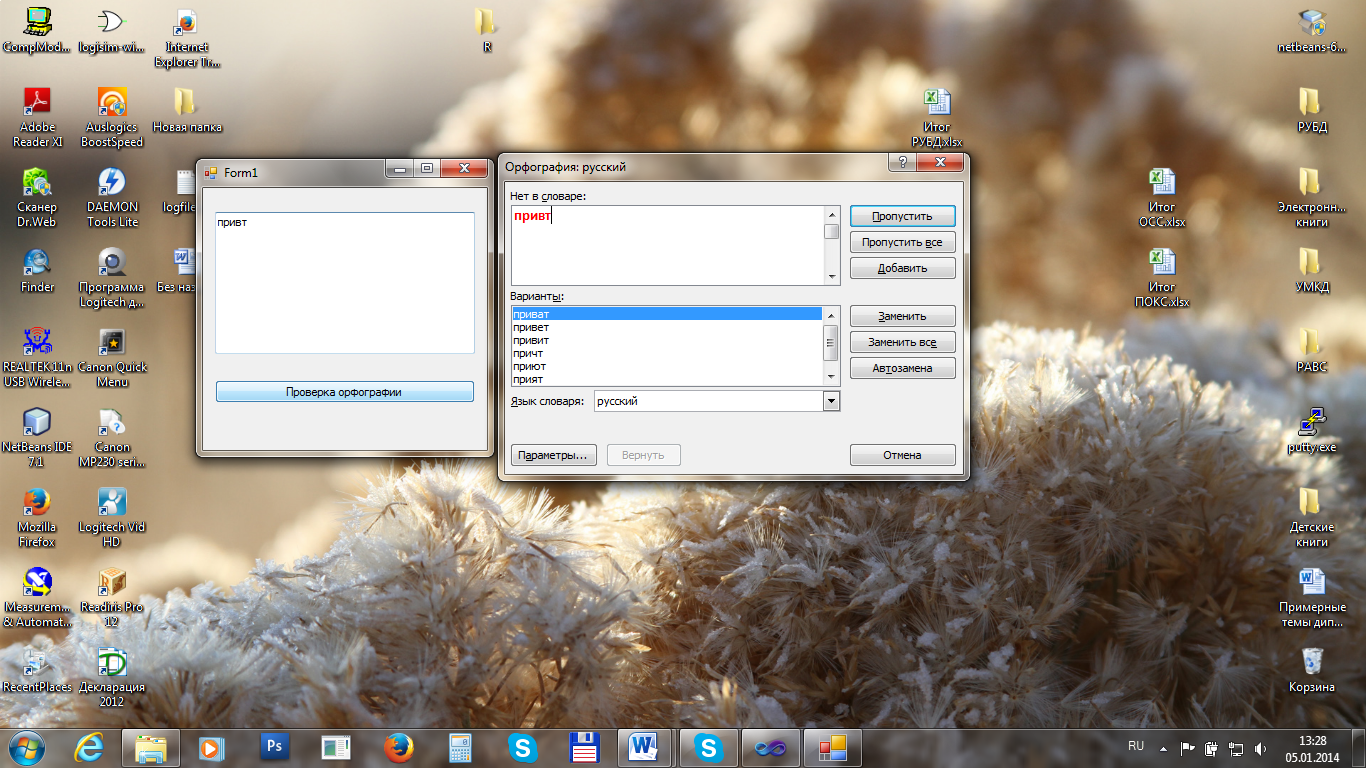


Рисунок 3 Результат работы программы

* + 1. Самостоятельно разберитесь с возможностью вывода в MSWord таблицы данных. Указания для выполнения: Опишите два массива, один массив – имена, второй – телефоны. Создайте новый объект Word1 для работы объектом с приложением MSWord. Свойству видимости объекта Word1 присвойте значение true. Добавьте новый документ в Word1. Добавьте текст «Таблица телефонов», используя свойство Selection.TypeText(*текст*) объекта Word1. Добавьте таблицу, состоящую из шести строк и двух столбцов. Организуйте цикл для перебора всех строк, в котором добавьте значения массивов, например так:   
        **Word1.ActiveDocument.Tables[1].Cell(i, 1).Range.InsertAfter(Names[i-1]);**
  1. Обращение к функциям MSExcel из программы на VisualC#.
     1. В текущее решение добавьте новый консольный проект.
     2. В проект добавьте ссылку на библиотеку объектов MSExcel. Для этого добавьте строчку **usingMicrosoft.Office.Interop.Excel;**
     3. Вразделе**static void Main(string[] args)** напишите программу:
        + Опишите XL1 как**new Application();**
        + Сделайте видимым данный объект;
        + Опишите переменную **t = Type.Missing** для задания значения по умолчанию и дальнейшего его использования в методах программы;
        + Создадим новую книгу MSExcel **– Book1=XL1.Workbooks.Add(t);**
        + Объявляем листы в книге **var Lists = Book1.Worksheets;**
        + Выбираем первый лист **Worksheet List1 = Lists.Item[1];**
        + Опишите массив месяцев **string[] Mas**, заполните его значениями месяцев;
        + Опишите массив целых чисел **Mas1**, для хранения продаж за описанные месяцы;
        + Добавьте в ячейку A1 значение – **List1.Range["A1", t].Value2 = "Месяцы";**
        + В ячейку B1 добавьте текст – "Продажи";
        + Организуйте цикл для просмотра всех значений массивов и добавления их значений в ячейки A2,…; B2,…
        + Напишите команды для добавления диаграммы и сохранения его в файле на диске, объясните назначение каждой команды:

**Chart График = XL1.Charts.Add(t, t, t, t);**

**График.ChartType = XlChartType.xlConeBarClustered;**

**График.SetSourceData(List1.Range["A1:C6"], t);**

**График.HasLegend = false;**

**График.HasTitle = true;**

**График.ChartTitle.Caption = "Продажи за пять месяцев";**

**XL1.ActiveChart.Export(@"E:\График.jpg", t, t);**

* + 1. Результат работы программы:

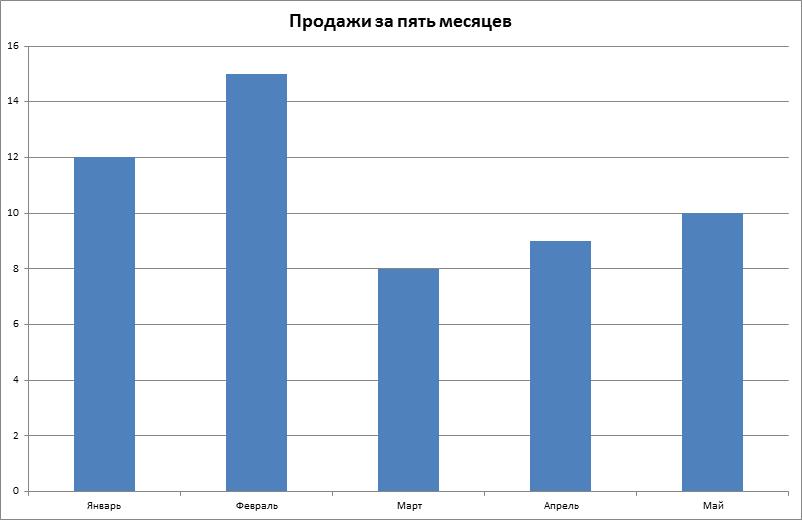


Рисунок 4 Полученная диаграмма

* + 1. Самостоятельно дополните проект следующими возможностями:
       - Добавьте в ячейку С1 значение – Налог;
       - Заполните значение колонки C значениями налога с продажи (18% умножьте на продажу);
       - Подведите итоги по столбцам B и C. Для этого: воспользуйтесь функцией **XL1.WorksheetFunction.Sum(List1.Range["B2:B6"]);**
       - Задайте для таблицы данных цвет границы по своему усмотрению;
       - Установите выбор источника данных для построения диаграммы – область List1.Range["A1:C6"];
       - Измените тип диаграммы по своему усмотрению, для этого измените свойство ChartType;

1. Содержание отчёта по практической работе
   1. Цель работы.
   2. Результаты выполнения работы;
   3. Выполненные работы в электронном виде представляются на сетевом диске H (X)
2. Контрольные вопросы
   1. Возможности использования функций MSWord.
   2. Возможности использования функций MSExcel.
3. Список рекомендуемой литературы:
   1. Агальцов В.П. Базы данных. Распределённые и удалённые базы данных. Учебник М.: ИД «Форум»: ИНФРА-М, 2011. - 272с.
   2. Голицына О.Л. Системы управления базами данных: учеб. пособие.- М.:ФОРУМ: ИНФРА-М,2006.- 432с
   3. Голощапов А.Л. MicrosoftVisualStudio 2010, - СПб.: БХВ-Петербург, 2011. – 544 с.
   4. Грекул В.И. Проектирование информационных систем. Курс лекций. Учебное пособие. М.: Интернет Ун-т Информ. технологий- 2005, 304с.
   5. Зиборов В.В. VisualC# 2012 на примерах. – СПб.: БХВ-Петербург, 2013. – 480 с.
   6. Культин Н.Б. MicrosoftVisualC# в задачах и примерах. – СПб.: БХВ-Петербург, 2013. – 320 с.
   7. Петров В.Н. Избачков Ю.С., Информационные системы. СПб.: Питер, 2006, 656с.
   8. Савицкий Н.М. Технологии организации, хранения и обработки данных: учеб. пособие.- М.:ФОРУМ: ИНФРА-М,2001.- 232с.
   9. Флеонов М.Е. Библия С#. – 2-е изд., перераб. и доп. СПб.: БХВ-Петербург, 2011. – 560 с.