Простая работа с MS Word

http://hiprog.com/index.php?option=com_content&task=view&id=379&Itemid=35

Автор: Юрий Шерман (16.04.2002 г.)

Вы вышли из кабинета шефа, вертя в руках причудливо разграфленный листок с названием, например, "Бланк - Заказ". Все ваши попытки убедить шефа, что его лучше печатать в виде отчета, окончились неудачей. "На Ворде, и тютелька в тютельку" - его последнее указание... В статье предлагается простое решение этой задачи

Практическая работа с Word

Хотя статья предназначена для начинающих, надеюсь, что и более опытные программисты найдут в ней кое-что полезное. Например, работу с Word без файлов на диске.

В качестве вступления.

Вы вышли из кабинета шефа, вертя в руках причудливо разграфленный листок с названием, например, "Бланк - Заказ".
Все ваши попытки убедить шефа, что его лучше печатать в виде отчета, окончились неудачей. "На Ворде, и тютелька в тютельку" - его последнее указание.
Сидя за столом и всматриваясь в листок, вы легко убеждаетесь, что "тютелька в тютельку" не получится. Просто слишком мало места отведено для названий (товара, услуг, и т.д.). И непонятно, как сделать подчеркивания под датами и подписями? Ведь эти подчеркивания явно придуманы для ручного вписывания текста, а для программы совсем не нужны...
Сгоряча собрав образцы, заполненные вручную вкривь и вкось, с записью ручкой прямо по надписям, с пренебрежением к подчеркнутым строкам, а то и просто на полях, вы готовы идти к шефу и доказывать, что этот листок вообще нельзя программно заполнить, его совершенно необходимо коренным образом переделать и ...
Но не спешите идти к шефу! На самом деле его все-таки можно сделать. И программой. Хотя не точно "в тютельку", но вполне приемлемо.
Как? Об этом эта статья. С примером.

Про "тютелька в тютельку", или как сделать файл-шаблон.

Точное выполнение "тютельки" - это создание файла Word (шаблона), распечатка которого на просвет совпадает с "Бланк - Заказом". Такое бывает нужно, но в другой постановке - точно впечатать нужную информацию в заданный бланк. Но это совершенно другая задача. Наша задача имеет к ней довольно отдаленное отношение. Для доказательства этого обратимся к реальной жизни. Рассмотрим, например, экземпляры официально утвержденного бланка "Счет - фактура", отпечатанного в разных типографиях. Мы легко заметим, что и строки, и колонки имеют слегка отличающиеся

размеры, могут применяться разные шрифты, даже в названиях колонок можно увидеть разные сокращения слов и т.д. То есть, не все так строго даже в официально утвержденных формах документов. Следовательно, и вам можно поиграть теми же параметрами, подгоняя "Бланк - Заказ" к своим потребностям. И шеф, как показывает практика, рассматривая готовый документ, не будет сравнивать его "на просвет" с исходным бланком. Так что, некоторую степень свободы вы имеете.

Однако надо выполнить основные требования:
- правильно расположить на листе названия и поля печати;
- сделать подчеркивание полей печати (за этим всегда строго следят);
- обеспечить возможность внесения данных большой длины без разрушения структуры документа.

Почти всегда бывают дополнительные требования:
- разные шрифты надписей и данных;
- различное выравнивание данных в отведенных полях.

И, наконец, обязательное свое требование: максимальную простоту программы формирования документа.

Как ни странно, но все эти достаточно разнородные задачи решаются одним приемом: повсеместным применением таблиц. Действительно:
- таблица с невидимыми границами ячеек обеспечит нужное размещение надписей и полей данных на странице;
- ячейка с видимой нижней границей подчеркнет поле данных;
- если текст велик, то ячейка автоматически увеличит свой размер вместе со всей строкой таблицы, сохраняя структуру документа;
- в разных ячейках можно установить любые шрифты и выравнивания;
- и, как будет показано ниже, передвижения по ячейкам и запись в них данных выполняются очень просто.
К тому же таблица обладает замечательным свойством: легко (программно) увеличить число ее строк, сохраняя заданное форматирование ячеек. Это используется при выводе каких-либо списков заранее неизвестной длины.

Вооружившись этим приемом, приступайте к рисованию файла-шаблона.
Маленькие рекомендации: для надписей хорош строгий шрифт Arial; для числовых данных подходит Courier для четкого выравнивания чисел по вертикали; для текстовых как Courier, так Times Roman. Жирным шрифтом лучше выделять данные, а не надписи.

Поэкспериментируйте с шаблоном, внося в него данные и распечатывая. Иногда на печати вид документа сильно отличается от вида на экране.

Заключительный файл-шаблон обычно имеет вид набора таблиц, с внесенными названиями и пустыми ячейками для данных. Таблицы отделяются абзацами неизменяемого текста. Помните, что все, что вы не внесли вручную в шаблон, придется потом вносить программно.

Если вы сразу получили в качестве образца файл на Word, то его все равно надо переработать к указанному здесь виду.

Конечно, все это носит характер рекомендаций. Правда, за этими рекомендациями стоят десятки созданных документов на Word.

Будем считать, что файл вы нарисовали.
Возникают вопросы:
- где его хранить?
- как на его основе создать файл для заполнения?
- как заносить данные?

Где хранить файл-шаблон, или об основной схеме работы.

Первое, что приходит в голову - хранить шаблон рядом с базой, затем создавать его копию, открывать и заполнять данными созданную копию.

Однако очевидно, что:
- файл-шаблон может быть легко испорчен или вовсе удален любопытным юзером;
- чтобы создать новую копию шаблона надо удалить предшествующую, а она может быть открыта;
- или сгенерировать новое имя копии, но тогда копии будут неограниченно размножаться и надо заботиться об их удалении;
- ...
То есть, перечень неприятностей при такой схеме работы можно продолжить, но и этих хватает, чтобы подумать о другой.

Основная идея другой схемы: раз работа с файлами приводит к неприятностям, то давайте обойдемся без файлов.

Для начала спрячем шаблон в таблицу. Таблицу назовем Files и сделаем в ней три поля:

- ID (число или текст, но не счетчик);
- Comment (текст) для описания для себя, что за файл лежит в этой записи;
- Shablon (OLE) для хранения шаблона.

Теперь откройте таблицу и запишите ID и Comment. Далее, используя вставку объекта, перепишите в поле Shablon созданный вами файл-шаблон.

Если вам потребуется откорректировать файл-шаблон, то его легко вызывать двойным щелчком по полю Shablon. То есть, будет вызван Word сразу с вашим файлом. Далее вы корректируете файл, и закрываете его. Изменения запишутся в таблицу.

Таблицу Files не следует использовать ни в каких формах. Тогда вероятность ее порчи любопытным юзером равна вероятности порчи любых других данных вашей базы. Различные методы защиты базы от любопытного юзера изложены в статье "Защита баз mdb".
Саму таблицу лучше хранить в клиентской части базы для ускорения загрузки файлов.

Как создать копию шаблона для заполнения данными.

Для этого в любой форме (я использую всегда открытую, невидимую форму LP) создайте присоединенную рамку объекта с именем, например, Shabl. Сделайте ее маленькой и невидимой. И ни к чему не присоединяйте. То есть поле "Данные" оставьте пустым. Установите свойство "Команда" (Verb) равным -2 (открывать объект в отдельном окне приложения). Стандартное значение, равное 0, тоже правильно, но иногда приводит к сбоям.

Для того, что бы скопировать в рамку файл-шаблон достаточно выполнить одну команду:

```
Forms!LP!Shabl = DlookUp("Shablon","Files","ID="&
Номер_файла)
```

Копировка сделана. Теперь нужно открыть копию. Это снова выполняется одной командой:

```
Forms!LP!Shabl.Action = acOLEActivate
```

При этом, если Word загружен не был, то он загружается и открывает копию. Если Word уже был загружен, то копия открывается в новом окне Word. При этом копия всегда

приобретает статус активного документа, что важно для программного доступа к ней. Курсор в открытом документе всегда стоит на первой позиции.
При этом подходе никакого анализа состояния Word делать не надо. Все делается автоматически и так, как нужно.

Далее вы (или юзер) можете изменять копию как угодно - она не связана с таблицей и исходный шаблон не изменится. Реально файла-копии на диске нет - копия сидит только в форме, то есть как бы висит в воздухе. Копия исчезнет при замене Forms!LP!Shabl на другой файл или при закрытии формы.
Но до этого вы можете заполнить копию данными, а юзер может исправить данные, отправить на печать или сохранить ваш воздушный файл в виде обычного, выполнив в Word команду "Сохранить копию как...".

Вот и вся схема работы с файлом-шаблоном без файлов на диске.
Два приятных замечания:
- в конце работы файл закрывать не нужно - пусть юзер делает с ним, что хочет - у вас от этого ничего не испортится;
- работу с Excel можно построить этим же способом.

Теперь осталось описать, как заполнить наш воздушный файл данными.

Как написать программу внесения данных.

1. Общее оформление программ.

Все программы работы с файлами Word следует поместить о один общий модуль. На уровне описания модуля укажите два объекта:

```
Dim obWord As Object 'это сам Word
Dim obWindow As Object 'окно нашего документа
```

Не делайте ссылки на библиотеку Word. Не используйте названий констант Word, а используйте их числовые значения. Как их получить - ниже. Это противоречит рекомендациям Help, но есть существенная причина поступать именно так: при переносе программы с компьютера на компьютер вы можете потерять ссылки и программа перестанет работать. Особенные неприятности возникают, если на компьютере установлены несколько Офисов разных версий. При отсутствии ссылок все работает. Правда при

вызове из A97 может вылезти Word-2000 даже при наличии Word-97. Но это ничему не мешает. Отлаженная программа работает в любых комбинациях.

2. Установление связи с Word и оформление его окна.

После активации вашего файла для установления связи с Word выполните:

```
Set obWord = GetObject(, "Word.Application")
Set obWindow = obWord.ActiveDocument.ActiveWindow
```

Далее максимизируйте окно документа и окно Word:

```
obWindow.WindowState = 1
obWord.WindowState = 1
```

Сделайте окно Word активным:

```
obWord.Activate
```

Конечно все можно сделать по-другому. (Например, спрятать окно Word.) Но именно к этому оформлению работы с окном Word привели следующие соображения:
- юзер может вызвать вашу программу по ошибке. И, если у него не будет доступа к Word, он не сможет остановить формирование документа. А оно может быть длительным. А так он кликнет по крестику и ваша программа вылетит по ошибке, потеряв связь с Word. Поставив реакцию на ошибку, вы сможете аккуратно прекратить формирование документа.
- юзеру обычно доставляет удовольствие видеть, как ваша программа заполняет поля шаблона. Не лишайте его этого удовольствия, заставляя его тупо глядеть на песочные часы.

3. Как написать программу работы с Word, ничего о нем не зная.

Работа с файлом Word складывается, естественно, из элементарных операций. Если вы умеете выполнить в Word нужную операцию, используя только клавиатуру, то Word может сам записать эту операцию в виде программы.

Для этого, установив курсор в файле Word в место, где, по-вашему, он должен находиться, включите запись макро (Сервис. Запись макро). Word выведет окно с приглашением указать имя макро и место его записи. Имя вам безразлично, но место обязательно укажите "Для данного

документа". По умолчанию Word пишет в Normal.dot, что вам совсем не нужно. После закрытия этого окна появится окошко управления записью макро с кнопками "Остановить запись" и "Пауза".
Дальше выполняйте нужную вам операцию. При записи макро действия мыши обычно блокированы. Поэтому операцию надо выполнять с клавиатуры. Но некоторые действия из меню доступны и мышью. Пробуйте. Закончив операцию, щелкните по кнопке "Остановить запись". Окошко закроется. Программа готова.

Для доступа к программе нажмите Alt+F8 (Сервис. Выполнить макро). Выберите в окне имя записанного макро и кликните по кнопке "Отладка". Word откроет текст макро в состоянии "выполнение" с остановкой на первой строчке. Скопируйте текст программы. Он пойдет в Access. Если в макро есть константы - посмотрите и запишите их значения. В программе Access замените константы на эти числа. Теперь почти все.

Надо еще указать в программе Access к каким объектам относятся скопированные команды. Word их не указывает. Обычно это obWindow, реже obWord. Это можно выяснить экспериментально, поставив перед командой один из объектов и запустив пошаговый режим отладки Access. Где нет ошибки, то и правильно.

Конечно, все изложенное никак не препятствует использованию вами Help по VBA для Word и любых других материалов.

Элементарные операции полезно оформить в виде крошечных подпрограмм с самоговорящими названиями. Из элементарных подпрограмм по мере необходимости можно создавать более сложные подпрограммы, выполняющие комплексные операции.

Такой же метод "программирования без программирования" можно использовать и при работе с Excel. Но у него есть свои особенности (и ошибки!), которые нужно описывать отдельно.

4. Навигация по файлу Word и запись данных.

Вся программа внесения данных состоит из передвижений курсора с целью попасть в нужную ячейку таблицы, и записи в нее очередного значения. Для записи достаточно одной команды. Поэтому сложность (весьма невысокая) состоит только в программировании перемещений курсора. Программа

легко отлаживается, так как все передвижения курсора по файлу видны в окне Word. Естественно, вносить данные в файл можно и по другому. Но этот способ - один из простейших.

Для программирования следует составить набор подпрограмм для перемещений курсора по строкам:
- в начало документа;
- в конец документа;
- вверх на n строк;
- вниз на n строк.

Навигация по таблице состоит из несколько большего набора подпрограмм, включающего движения по ячейкам влево и вправо. Движение вправо из последней ячейки таблицы создает новую строку таблицы.

Эти подпрограммы вы можете составить сами, пользуясь методом "программирования без программирования". И только потом сверить их с текстами, данными в примере.

Надо отметить, что возможности Word очень многообразны. Например, для навигации можно использовать закладки и другие средства, а не только относительные перемещения курсора. Следует расширять набор подпрограмм, выполняющих новые операции, по мере необходимости.

О примере.

Пример написан для A-97 в формате mdb. Его можно конвертировать в A-2000. В примере одна форма, из которой вызываются программы создания двух документов:
- приходного ордера, с использованием только методов, изложенных выше;
- списка людей со шапкой, взятой из другого файла Word.
Второй случай посложнее. Он демонстрирует слияние двух файлов через ClipBoard, чтение данных из файла и некоторое другое. Комментарии к этому случаю даны в самой программе.

Послесловие.

Здесь и в примере даны только простейшие средства, которых, однако хватает для создания большинства документов. Надеюсь, что статья пригодится начинающим для "сдвига с мертвой точки" в программном создании файлов Word.

Возвращаясь к вступлению.
Через некоторое время вы вдруг осознаете, что шеф был прав, настаивая на Ворде. С ним исчезли многие проблемы, связанные с применением отчетов. И главная из них - не надо больше в авральном порядке изменять отчеты. Для начала юзер сам поправит в файле Ворда все, что нужно. А вы потом запрограммируете только устоявшиеся изменения. Другая - вам больше незачем заботиться о том, чтобы получатель мог просмотреть файл, посланный по почте. Word у него гарантированно есть.