ДМИТРИЙ КУЗАН
ВЛАДИМИР ШАПОРОВ

bhv

ПРОГРАММИРОВАНИЕ Win32 API В DELPHI

ВВЕДЕНИЕ В Win32 API

GDI+ – ИНТЕРФЕЙС НОВОГО ПОКОЛЕНИЯ

SIMPLE MAPI – РАБОТА С ЭЛЕКТРОННОЙ ПОЧТОЙ

VIDEO FOR WINDOWS – РАБОТА С ВИДЕО

MEDIA CONTROL INTERFACE – РАБОТА С МУЛЬТИМЕДИА

TAPI – РАБОТА СО СРЕДСТВАМИ КОММУНИКАЦИЙ

PRO
ПРОФЕССИОНАЛЬНОЕ ПРОГРАММИРОВАНИЕ

+CD

Дмитрий Кузан
Владимир Шапоров

ПРОГРАММИРОВАНИЕ Win32 API в DELPHI

Санкт-Петербург
«БХВ-Петербург»
2005

УДК 681.3.06
ББК 32.973.26-018.2
К89

Кузан Д. Я., Шапоров В. Н.

К89 Программирование Win32 API в Delphi. — СПб.: БХВ-Петербург, 2005. — 368 с.: ил.

ISBN 5-94157-535-1

Рассмотрено применение различных интерфейсов прикладного программирования Windows (Win32 API) при разработке приложений с использованием Borland Delphi. Описаны основы работы с API. Подробно освещены вопросы практического применения API при создании приложений для работы с электронной почтой (MAPI), со средствами коммуникаций (TAPI), мультимедиа (MMCI), графическим интерфейсом и др. Материал сопровождается наглядными практическими примерами. На компакт-диске расположены исходные тексты примеров, программы и необходимые библиотеки.

Для программистов

УДК 681.3.06
ББК 32.973.26-018.2

Группа подготовки издания:

Главный редактор	*Екатерина Кондукова*
Зам. главного редактора	*Игорь Шишигин*
Зав. редакцией	*Григорий Добин*
Редактор	*Татьяна Лапина*
Компьютерная верстка	*Ольги Сергиенко*
Корректор	*Зинаида Дмитриева*
Дизайн серии	*Инны Тачиной*
Оформление обложки	*Игоря Цырульникова*
Зав. производством	*Николай Тверских*

Лицензия ИД № 02429 от 24.07.00. Подписано в печать 22.09.05.
Формат 70×100 1/16. Печать офсетная. Усл. печ. л. 29,67.
Тираж 3000 экз. Заказ №
"БХВ-Петербург", 194354, Санкт-Петербург, ул. Есенина, 5Б.

Санитарно-эпидемиологическое заключение на продукцию
№ 77.99.02.953.Д.006421.11.04 от 11.11.2004 г. выдано Федеральной службой
по надзору в сфере защиты прав потребителей и благополучия человека.

Отпечатано с готовых диапозитивов
в ГУП "Типография "Наука"
199034, Санкт-Петербург, 9 линия, 12

ISBN 5-94157-535-1

ы

Оглавление

Введение 8

Глава 1. MAPI – интерфейс программирования приложений электронных сообщений 9

Введение 9
Достоинства и недостатки Simple MAPI 11
Подключение Simple MAPI к проекту 12
Отправка сообщения на Simple MAPI 12
Работа с адресной книгой на Simple MAPI 23
Работа с сообщениями на Simple MAPI 28
Коды ошибок Simple MAPI 33

Глава 2. TAPI – интерфейс программирования приложений для работы с телефонией 36

Введение в TAPI 36
Интерфейсы и уровни программирования TAPI 37
Базовый уровень 38
Вспомогательный уровень 39
Расширенный уровень 40
Работа с устройствами линий 40
Основные шаги работы с телефонией 40
Конфигурирование и настройка устройства коммуникации 41
Структура *VarString TAPI* 42
Три механизма уведомлений (сообщений) TAPI 44
Версионность TAPI 46
Определение способностей телефонии 47
Открытие устройства линии 49
Дайте мне ваш ID 50
Базовые функции TAPI 50
Вспомогательные функции TAPI 54

Обработка сообщений линии TAPI....58
LineCallback — функция обработки сообщений линии....59
Сообщения линии TAPI....60
Порядок поступления сообщений для входящих и исходящих вызовов....65
Функции и структуры TAPI, связанные с обработкой сообщений....66
Размещение исходящих вызовов TAPI....71
Форматы номеров телефонов в TAPI....71
Ассистент телефонии....73
Функции ассистента телефонии....74
Установление вызова с помощью низкоуровневых функций линии....75
Принятие входящих вызовов....93
Поиск заинтересованного приложения....93
Неизвестный режим носителей....95
Приоритет режимов носителей....96
Обязанности приложения, принимающего входящие вызовы....97
Регламент работы приложения, определяющего режим носителей....98
Принятие входящего вызова....100
Завершение вызова....102
Функции и структуры TAPI, управляющие приемом вызовов....105
Заключение....117

Глава 3. MCI-интерфейс для работы с мультимедиа....118

Введение....118
Интерфейс командных строк и команд-сообщений MCI....118
Командные строки....119
Команды сообщений....120
Типы и драйверы MCI-устройств....121
Классификация MCI-команд....123
Функции и макросы MCI....128
Сообщения MCI....130
Общие флаги для MCI-команд....133
Структуры данных MCI....133
Практика использования....141
Проигрывание wave-файлов....142
Проигрывание MIDI-файлов....147
Звукозапись....150
Проигрывание Audio-CD....153
Проигрывание видеофайлов AVI....154
Коды ошибок MCI....157
Заключение....162

Глава 4. Video for Windows – интерфейс для работы с видео....163

Краткий экскурс....163
Введение в Video for Windows....164
Установка и требования к работе....164

Практика использования ... 165
Открытие файлов AVI ... 165
Получение информации из заголовка файла AVI ... 167
Доступ к потокам ... 170
Получение информации о потоке ... 171
Работа с кадрами. Сохранение отдельных кадров в формат BMP ... 184
Работа с кадрами. Сохранение BMP-файлов в AVI-формат ... 188
Сохранение потоков в отдельных файлах ... 198
Обработка ошибок VFW ... 203
Заключение ... 205

Глава 5. GDI+ — графический интерфейс нового поколения ... 206

Введение в GDI+ ... 206
Установка и требования к работе ... 207
Объектная модель библиотеки ... 208
Первые шаги ... 209
Классы GDI ... 212
Класс *AdjustableArrowCap* ... 212
Класс *Bitmap* ... 213
Класс *BitmapData* ... 213
Класс *Brush* ... 213
Класс *CachedBitmap* ... 213
Класс *CharasterRange* ... 213
Класс *Color* ... 214
Класс *CustomLineCap* ... 214
Класс *EncoderParametr* ... 214
Класс *EncoderParametrs* ... 215
Класс *Font* ... 215
Класс *FontCollection* ... 215
Класс *FontFamily* ... 215
Класс *GDIPlusBase* ... 215
Класс *Graphics* ... 215
Класс *GraphicsPath* ... 216
Класс *GraphicsPathIterator* ... 216
Класс *HatchBrush* ... 216
Класс *Image* ... 217
Класс *ImageAttributes* ... 217
Класс *ImageCodecInfo* ... 217
Класс *InstalledFontCollection* ... 217
Класс *LinearGradientBrush* ... 218
Класс *Matrix* ... 218
Класс *Metafile* ... 218
Класс *MetafileHeader* ... 218
Класс *PathData* ... 218
Класс *PathGradientBrush* ... 218

Класс *Pen* 219
Класс *Point* 219
Класс *PointF* 219
Класс *PrivateFontCollection* 219
Класс *PropertyItem* 219
Класс *Rect* 220
Класс *RectF* 220
Класс *Region* 220
Класс *Size* 220
Класс *SizeF* 220
Класс *SolidBrush* 220
Класс *StringFormat* 220
Класс *TextureBrush* 221
Перечисления GDI+ 221
Константы и структуры GDI+ 222
Практика использования 222
Рисование графических примитивов 222
Работа с изображениями 226
Использование кэшированных растров для повышения производительности вывода 237
Использование кодеров и декодеров изображений 239
Работа со списком кодеков 239
Получение CLSID кодера изображения 240
Определение параметров кодера 241
Сохранение изображений 243
Работа с метаданными 249
Использование Alpha-канала для создания эффектов прозрачности 251
Работа с текстом 254
Координатная система 267
Преобразования (трансформации) объектов 271
Использование регионов 280
Печать 282
Заключение 285

Глава 6. Windows API 286

Типы данных 286
Константы 290
Строки 290
Дескрипторы 292
Сообщения 293
Синтаксис функций Windows API 296
Параметры функций 297
Импортирование функций Windows API 298
Нестандартно импортируемые функции 298
Функции обратного вызова 299

Использование справочной системы по функциям Windows API.....300
Delphi и функции API.....301
Функции управления окнами.....302
Функции ввода/вывода в файл.....314
Функции ввода.....331
Строковые функции и функции атомов.....337
Функции работы с буфером обмена.....343
Функции системной информации.....347
Функции каретки, курсора и иконок.....355
Заключение.....362

Приложение. Описание компакт-диска..... 363

Предметный указатель..... 364

Введение

Уважаемый читатель, вы держите в руках книгу, рассказывающую о взаимодействии Delphi с различными интерфейсами прикладного программирования (API). В ней мы постарались ознакомить вас с различными API и показать работу с ними в Delphi. Нами не ставилась задача подробно и полностью описать каждый интерфейс прикладного программирования, да и это было бы невозможно в рамках одной книги. Скорее всего, ее можно было бы охарактеризовать как вводящую читателя в мир API. Не секрет, что в настоящее время существует огромное количество книг по Delphi, но, к сожалению, большая часть из них посвящена вопросам визуального программирования. Очень часто в них встречаются фразы типа "возьмите компонент Tlabel из палитры компонентов и положите его на форму". Авторы ни в коем случае не против таких книг, но практика показывает, что многие из них просто дублируют друг друга. Стоит также упомянуть, что большинство из них предназначено для новичков. Мы же в свою очередь постарались отойти немного в сторону от стандартов и выбрали тему взаимодействия Delphi с различными API не случайно. Дело в том, что в настоящее время мало материалов, посвященных Delphi и API, к тому же большая их часть опубликована на английском языке, и этой книгой мы постарались хоть как-то заполнить вакуум, присутствующий в компьютерной литературе. Мы прекрасно понимаем, что она вряд ли будет служить полным руководством для разработчиков, использующих соответствующие API, но она может стать, и надеемся, станет первым шагом, который заставит программистов, использующих Delphi, начать изучать различные API. Ведь на Delphi можно создавать не только различные оконные приложения и приложения для работы с базами данных, но и сложные профессиональные приложения, например, для работы со звуком, видео или электронными коммуникациями. Тем более что такие приложения в большинстве используют сторонние API. И каждый уважающий себя программист должен если не знать, то хотя бы разбираться в многообразии различных API.

Итак, мы надеемся, что данная книга послужит путеводителем по различным API для программистов и даст толчок для усовершенствования знаний в программировании. Удачи вам!

ГЛАВА 1

MAPI – интерфейс программирования приложений электронных сообщений

Введение

С каждым днем электронными коммуникациями пользуются все больше и больше людей. Электронный почтовый ящик уже не вызывает удивления, как некоторое время назад; напротив, в компьютерном сообществе все меньше и меньше остается людей, не имеющих электронного почтового ящика. Эта тенденция продолжает сохраняться, поэтому перед программистами встают все новые и новые задачи. Если раньше заказчику требовался просто отчет из базы данных, то теперь заказчик уже хочет, чтоб этот отчет отправлялся ему на дом автоматически по электронной почте. Время не стоит на месте, и смогут ли разработчики не впасть в уныние от множества запутанных стандартов API, которые определяют для себя системы электронных коммуникаций. Фактически основными протоколами передачи электронных почтовых сообщений стали SMTP и POP3, и современный разработчик должен в них разбираться. Тут знающий читатель может задать вопрос: "А зачем мне разбираться с этими протоколами, если в Delphi есть для этого необходимые компоненты, кинув которые на форму я могу придать своему приложению необходимую функциональность?". Это, безусловно, так. Однако на этом пути есть пара, на наш взгляд, очевидных минусов.

Перечислим некоторые из них.

- *Необходимая минимальная настройка* — попробуем объяснить, что мы под этим подразумеваем. Несмотря на все преимущества этих компонентов, их необходимо настроить, т. е. в вашем приложении предусмотреть возможность хранения таких настроек, как электронный адрес, пароль на ящик и пр. Если вы разрабатываете одновременно несколько приложений, использующих электронную почту, то трудоемкость программирования резко возрастает.

- *Отсутствие встроенных средств протоколирования* — один из самых больших недостатков компонентов. Предположим, вы разрабатываете приложение, позволяющее отправлять индивидуальные письма большому количеству получателей автоматически, и вам необходимо предусмотреть возможность ведения архива отправленных сообщений. При определенном опыте можно написать собственные средства ведения логов, однако согласитесь, если бы они были встроены заранее, жить стало бы намного легче.
- *Встроенные ошибки* — несмотря на тщательное тестирование этих компонентов, в них могут быть заложены ошибки. Так сами авторы столкнулись с тем, что стандартные компоненты NMSMTP и NMPOP3, расположенные на вкладке FastNet среды Delphi, не всегда ведут себя корректно. В частности, NMPOP3 не очень хорошо работает с кодировкой KOI8-R.
- *Изменчивость стандартов* — несмотря на довольно широкое распространение стандартов, нет никакой гарантии, что эти стандарты не изменятся в будущем. Соответственно может настать время, когда ваше приложение просто перестанет работать. Хоть это и маловероятно, но возможность существует.

А теперь представим, что в ваши руки попал инструмент электронных сообщений, практически не ограниченный данными минусами. Используя его, вы навсегда забудете заботы о совместимости протоколов и стандартов, все неудобства, связанные с компонентами. Имя этому средству — MAPI или Messaging API. MAPI — интерфейс программирования приложений электронных сообщений предоставит в ваши руки надежный инструмент работы с почтовыми сообщениями. Вкратце мы попробуем обрисовать, что нам может дать использование MAPI в наших приложениях.

- *Простой способ отправки и приема почты* независимо от установленного почтового клиента. Настройки соединения и подключения берутся из самого почтового клиента. Единственное ограничение: почтовый клиент должен поддерживать стандарт MAPI. Хотя ограничением это назвать можно с натяжкой — популярные Outlook Express, Microsoft Outlook и горячо любимый многими The Bat давно уже поддерживают данный стандарт.
- *Протоколирование* — все отправленные письма будут автоматически ложиться в папку "Отправленные" вашего почтового клиента.
- Отсутствие проблем с национальной кодировкой.
- Простой доступ к общей адресной книге. Теперь не надо вести адресную книгу в самих приложениях.
- Гарантия совместимости в будущем.

И это далеко не полный список тех достоинств, что дает нам MAPI. Однако прежде чем приступить непосредственно к разбору процедур и функций, остановимся на архитектуре MAPI. Прежде всего, запомним, что MAPI не является традиционным API, хотя MAPI и представляет разработчикам необходимые функции. Но это только верхняя часть айсберга! На самом деле *MAPI — это архитектура, предоставляющая спецификацию для всех систем электронных сообщений*.

MAPI — это архитектура, основанная на отраслевом стандарте, она не является чьей-либо собственностью, и разрабатывается по сей день множеством различных производителей программ (ISV — independent software vendors), желающих добавить системе электронных сообщений некоторые функции.

В данной главе мы наиболее подробно остановимся на простом варианте MAPI, выпущенном фирмой Microsoft. Он называется Simple MAPI (простой MAPI) и представляет собой обертку для вызовов простейших функций управления почтовыми сообщениями. Simple MAPI позволяет выполнить самые основные действия по рассылке и приему почтовых сообщений. В Simple MAPI входят 12 основных функций и несколько структур данных.

В данной главе мы не будем останавливаться на более позднем варианте MAPI — MAPI 1.0 или Extended MAPI (расширенном MAPI), кардинально отличающемся от своего предшественника. Подробное рассмотрение MAPI 1.0 вылилось бы в написание отдельной книги, и поэтому мы решили остановиться на Simple MAPI, как на наиболее простом в изучении и понимании. Для тех же, кому нужно получить информацию по использованию MAPI 1.0, а также заголовочные файлы и примеры, мы можем посоветовать обратиться на соответствующие сайты в Интернете, в частности на **www.iCodeTeam.net, www.evocorp.com**.

Однако прежде чем приступить к рассмотрению Simple MAPI, мы немного расскажем о MAPI 1.0. В отличие от Simple MAPI, потомок MAPI 1.0 предоставляет большую функциональность вкупе с широкими возможностями. Основанный на технологии COM данный MAPI 1.0 представляет разработчикам большой спектр функций и COM-интерфейсов для создания приложений электронных коммуникаций. Более сложный в освоении MAPI 1.0 на самом деле является мощным инструментом, позволяющим создавать такие части почтовых систем, как, например, почтовые шлюзы и даже почтовые серверы.

Достоинства и недостатки Simple MAPI

Как вы уже знаете, Simple MAPI предоставляет разработчику простой набор средств для работы с почтой. Данный набор средств идеально подходит для создания небольших прикладных приложений, включающих в себя стандарт-

ные средства работы с почтой. Однако использование Simple MAPI накладывает на разработчика и ряд ограничений, не позволяющих создавать сложные почтовые системы.

- Разработчик не имеет возможности манипулировать сообщениями, находящимися за границами стандартных папок "Входящие", "Исходящие". То есть если вы хотите получать, например, доступ к папке "Письма для Васи", то Simple MAPI такой возможности вам не предоставит.
- Simple MAPI работает только со стандартными полями сообщений, такими как "Тема", "Отправитель", "Получатель" и т. п.
- Simple MAPI представляет собой интерфейс для отправки и приема сообщений и ни на что другое не способен.

Однако Simple MAPI дает простую и четкую возможность с минимальными усилиями встроить поддержку почтовых сообщений в собственное приложение.

Подключение Simple MAPI к проекту

Для того чтобы разработчик Delphi получил в свои руки все возможности Simple MAPI, ему необходимо подключить в секцию `Uses` файл `Mapi.pas`. Данный файл представляет собой заголовочный файл `Mapi.h`, переведенный в синтаксис Object Pascal компанией Borland.

Отправка сообщения на Simple MAPI

Прежде чем приступить к детальному освещению вопросов, касающихся использования Simple MAPI, приведем небольшой пример, позволяющий отправлять почтовые сообщения одному адресату через почтовый клиент пользователя с использованием Simple MAPI. На рис. 1.1 представлен внешний вид демонстрационного приложения, а в листинге 1.1 — его исходный код. Полную версию программы вы сможете найти на прилагаемом компакт-диске в каталоге Source\Ch01\Ex01.

Листинг 1.1. Отправка почтового сообщения с помощью Simple MAPI

```
procedure TFormMain.SendEMail;
var
  MapiMessage: TMAPIMessage;     // Структура содержит информацию
                                 // о посылаемом или
                                 // принимаемом сообщении.
  Receip     : TMAPIRecipDesc;  // Структура получателей.
```

```
  MAPI_Session: Cardinal;       // Сессия
  Flag       : Integer;
  dwRet      : Cardinal;        // Возвращаемое значение
  WndList: Pointer;
begin
  // Открываем сессию Simple MAPI
  dwRet := MapiLogon(Application.Handle,PChar(''),PChar(''),
                   MAPI_LOGON_UI or MAPI_NEW_SESSION,0,@MAPI_Session);
  // Если процесс установки почтовой сессии завершился неудачей
  if (dwRet <> SUCCESS_SUCCESS) then
  begin
      ShowMessage('Не могу установить сессию !');
  end
  else
  begin
     // Обнуляем структуру сообщения и структуру получателей.
     FillChar(MapiMessage, SizeOf(MapiMessage), #0);
     FillChar(Receip, SizeOf(Receip), #0);
     // Заполняем поле "Получатель"
     Receip.ulReserved   := 0;
     Receip.ulRecipClass := MAPI_TO;  // Кому
     Receip.lpszName     := StrNew(PChar(EditTo.Text));
     Receip.lpszAddress  := StrNew(PChar('SMTP:' + EditTo.Text));
     Receip.ulEIDSize := 0;
     MapiMessage.nRecipCount := 1; // Кол-во получателей - один
     // Связываем получателей с сообщением
     MapiMessage.lpRecips := @Receip;
     // Вложенных файлов в сообщении пока нет
     MapiMessage.nFileCount := 0;
     MapiMessage.lpFiles    := nil;
     // Заполняем поле "Тема сообщения"
     if EditSubject.Text <> '' then
        MapiMessage.lpszSubject := StrNew(PChar(EditSubject.Text));
     // Заполняем сам текст сообщения
     if MemoBody.Text <> '' then
        MapiMessage.lpszNoteText := StrNew(PChar(MemoBody.Text));
     WndList := DisableTaskWindows(0);
     try
       IF CheckBoxSend.Checked then
          Flag := 0
       else
          Flag := MAPI_DIALOG; // Показывать стандартное окно почтовика
          MapiSendMail(MAPI_Session, Application.Handle,
                       MapiMessage, Flag, 0);
```

```
    finally
      EnableTaskWindows( WndList );
    end;
    // Очищаем выделенную память
    if Assigned(MapiMessage.lpszSubject) then
       StrDispose(MapiMessage.lpszSubject);
    if Assigned(MapiMessage.lpszNoteText) then
       StrDispose(MapiMessage.lpszNoteText);
    if Assigned(Receip.lpszAddress) then
       StrDispose(Receip.lpszAddress);
     if Assigned(Receip.lpszName) then
       StrDispose(Receip.lpszName);
     // Завершаем сессию
     MapiLogOff(MAPI_Session, Application.Handle, 0, 0);
  end;
end;
```

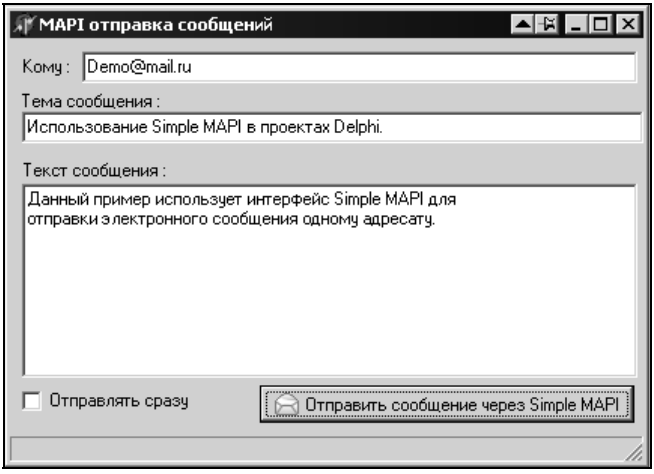

Рис. 1.1. Отправка сообщения через Simple MAPI

Как вы сами видите, для того чтобы отправить сообщение средствами Simple MAPI нужно вызвать всего несколько функций данного интерфейса, а достигнутая функциональность с лихвой окупает затраты на написание исходного кода. После того как мы с вами рассмотрели демонстрационный пример, углубимся в теоретическую часть и для начала определимся с сессиями Simple MAPI.

Концепция работы интерфейса Simple MAPI и прикладной программы основана на использовании сессий или, иначе говоря, соединений между прикладной программой и низкоуровневой системой электронных сообщений.

То есть перед непосредственным использованием функций Simple MAPI по работе с почтовыми сообщениями вы должны установить сессию (сеанс) работы с Simple MAPI с помощью функции `MAPILogon`. Установка сессии происходит в момент регистрации пользователя, когда операционная система убеждается в необходимых правомочиях системы сообщений, а также в отсутствии каких-либо ошибок со стороны профилей службы сообщений.

При установке сессии программист может указать один из двух типов сессии. Сессии могут быть как индивидуальные, так и разделяемые. Индивидуальные сессии являются соединениями "один к одному" между конкретным приложением и Simple MAPI, и не могут быть использованы другими внешними приложениями. Разделяемые сессии, напротив, могут быть использованы сторонними приложениями.

Для создания новой сессии, первичной загрузки адресной книги и списка сообщений нужно использовать функцию `MAPILogOn`. Ее прототип в Object Pascal имеет следующий вид:

```
function MapiLogOn(ulUIParam: Cardinal; lpszProfileName: LPSTR;
          lpszPassword: LPSTR; flFlags: FLAGS;
          ulReserved: Cardinal;
          lplhSession: PLHANDLE):Cardinal;
```

Данная функция имеет следующие параметры, представленные в табл. 1.1.

Таблица 1.1. *Параметры функции* `MAPILogOn`

Параметр	Описание
`ulUIParam`	Дескриптор окна, вызвавшего `MAPILogOn`. Если указан 0, то параметр `ulUIParam` игнорируется
`lpszProfileName`	Указатель на строку, содержащую путь к файлу настроек
`lpszPassword`	Указатель на строку, содержащую пароль для доступа
`flFlags`	Маска флагов дополнительных опций. В данном параметре могут быть установлены следующие флаги: `MAPI_FORCE_DOWNLOAD` — если флаг установлен, то функция пытается загрузить все сообщения на почтовом ящике перед окончанием работы данной функции. Если флаг не установлен, тогда сообщения будут загружены после завершения работы функции; `MAPI_LOGON_UI` — показывает диалог идентификации пользователя, при затребовании системой; `MAPI_NEW_SESSION` — флаг указывает функции сделать попытку создания новой сессии, перед тем как использовать существующую

Таблица 1.1 *(окончание)*

Параметр	Описание
ulReserved	Данный параметр не используется и зарезервирован на будущее
lplhSession	Указатель на сессию Simple MAPI. При создании новой сессии в данный параметр заносится указатель на созданную сессию

После создания сессии все следующие операции с почтовыми сообщениями используют идентификатор сессии, заданный в параметре `lplhSession`. Стоит также отметить, что после окончания работы с Simple MAPI следует закрыть открытую сессию, используя функцию Simple MAPI — `MAPILogOff`:

```
function MapiLogOff(lhSession: LHANDLE; ulUIParam: Cardinal;
          flFlags: FLAGS;
          ulReserved: Cardinal): Cardinal;
```

Данная функция закрывает открытую сессию с почтовой системой и имеет следующие параметры, представленные в табл. 1.2.

Таблица 1.2. *Параметры функции* `MAPILogOff`

Параметр	Описание
lhSession	Указатель на открытую ранее сессию Simple MAPI
ulUIParam	Дескриптор окна, вызвавшего `MAPILogOff`. Если указан 0, то данный параметр игнорируется

Параметры `flFlags` и `ulReserved` не используются и зарезервированы на будущее.

Следующим шагом после открытия сессии стал шаг заполнения полей структуры `TMAPIMessage`, содержащей информацию о посылаемом или принимаемом сообщении. Данная структура в Mapi.pas представлена следующим образом:

```
PMapiMessage = ^TMAPIMessage;
{$EXTERNALSYM MapiMessage}
MapiMessage = packed record
 ulReserved: Cardinal;
  lpszSubject: LPSTR;
  lpszNoteText: LPSTR;
  lpszMessageType: LPSTR;
  lpszDateReceived: LPSTR;
  lpszConversationID: LPSTR;
```

```
  flFlags: FLAGS;
  lpOriginator: PMapiRecipDesc;
  nRecipCount: Cardinal;
  lpRecips: PmapiRecipDesc;
  nFileCount: Cardinal;
  lpFiles: PMapiFileDesc;
 end;
TMAPIMessage = MapiMessage;
```

Структура имеет следующие поля, представленные в табл. 1.3.

Таблица 1.3. *Поля структуры* `TMAPIMessage`

Поле	Описание
`ulReserved`	Поле зарезервировано. Значение поля всегда установлено в ноль
`lpszSubject`	Указатель на строку, содержащую тему сообщения
`lpszNoteText`	Указатель на строку текста сообщения
`lpszMessageType`	Указатель на строку, указывающую тип сообщения
`lpszDateReceived`	Указатель на дату сообщения в формате YYYY/MM/DD HH:MM, где HH (часы) — имеет диапазон от 0 до 24. Данный параметр используется при получении сообщений
`lpszConversationID`	Указатель на идентификатор потока, в который поступило сообщение. Данный параметр используется при получении сообщений
`flFlags`	Маска флагов дополнительных опций сообщения, в данном параметре могут быть установлены следующие флаги: `MAPI_RECEIPT_REQUESTED` — запрашивать подтверждение получения; `MAPI_SENT` — флаг, указывающий на то, что сообщение было послано; `MAPI_UNREAD` — флаг, указывающий на то, что сообщение не было прочитано
`lpOriginator`	Информация об отправителе сообщения
`nRecipCount`	Количество получателей сообщения, записанных в массиве `lpRecips`
`lpRecips`	Указатель на массив структур `TMAPIRecipDesc`, содержащих в свою очередь информацию о получателях сообщения
`nFileCount`	Количество вложенных в сообщение файлов, содержащихся в массиве `lpFiles`

Таблица 1.3 *(окончание)*

Поле	Описание
lpFiles	Указатель на структуру, содержащую информацию обо всех файлах, приложенных к сообщению

Наконец, после того как заполнили структуру TMAPIMessage, мы можем выполнить непосредственно отправку сообщения, используя функцию MAPISendMail:

```
function MapiSendMail(lhSession: LHANDLE; ulUIParam: Cardinal;
          var lpMessage: TMAPIMessage;
          flFlags: FLAGS;
          ulReserved: Cardinal): Cardinal;
```

Данная функция производит отправку сообщения. Достоинством данной функции является то, что она позволяет производить отправку сообщения с вложенными в него файлами или OLE-объектами, что выгодно отличает ее от аналогичной функции Simple MAPI MAPISendDocuments, которая будет рассмотрена далее. Можно сказать, что MAPISendMail является более мощным вариантом функций отправки сообщений Simple MAPI.

Функция MAPISendMail имеет следующие параметры, представленные в табл. 1.4.

Таблица 1.4. *Параметры функции* MAPISendMail

Параметр	Описание
lhSession	Указатель на открытую ранее сессию Simple MAPI
ulUIParam	Дескриптор окна, вызвавшего MAPILogOff. Если указан 0, то данный параметр игнорируется
lpMessage	Указатель на структуру TMAPIMessage, содержащую сообщение для посылки
flFlags	Маска флагов дополнительных опций отправки. В данном параметре могут быть установлены следующие флаги: MAPI_DIALOG — показывает, что будет выведено стандартное окно отправки сообщения почтового клиента; MAPI_LOGON_UI — показывает диалог идентификации пользователя, при затребовании системой; MAPI_NEW_SESSION — флаг указывает функции сделать попытку создания новой сессии, перед тем как использовать существующую
ulReserved	Параметр зарезервирован

Более простой, но в то же время ограниченной альтернативой функции MAPISendMail является функция MAPISendDocuments, которая позволяет подготовить письмо и вложить в него файлы без указания адресов получателей. При использовании данной функции всегда необходимо участие пользователя:

```
function MapiSendDocuments(ulUIParam: Cardinal;
              lpszDelimChar: LPSTR;
              lpszFilePaths: LPSTR;
              lpszFileNames: LPSTR;
              ulReserved: Cardinal): Cardinal;
```

Операционная система использует именно эту функцию для подготовки письма к отправке при вызове контекстного меню "Отправить>Адресат". Функция имеет следующие параметры, представленные в табл. 1.5.

Таблица 1.5. *Параметры функции* `MAPISendDocuments`

Параметр	Описание
ulUIParam	Дескриптор окна, вызвавшего MAPILogOff. Если указан 0, то данный параметр игнорируется
lpszDelimChar	Указатель на разделитель файлов
lpszFullPaths	Указатель на строку, содержащую полные пути до файлов, которые будут вложены в отправляемое сообщение. При наличии нескольких отправляемых файлов имена файлов должны быть разделены знаком, заданным в параметре lpszDelimChar
lpszFileNames	Указатель на строку, содержащую оригинальные названия файлов, под которыми они будут видны в сообщении
ulReserved	Параметр зарезервирован

Пример подготовки двух документов на отправку:

```
MapiSendDocuments(0,';','d:\Письмо.doc;d:\Договор.doc',
          'Письмо.doc;Договор.doc',0);
```

После того как мы разобрали основные вопросы, касающиеся отправки сообщения, переработаем немного наш первоначальный пример, добавив в него возможность выбора нескольких получателей, а также встроив в наше приложение возможность прикрепления файлов к сообщению.

Непосредственно для реализации нам нужно подробно ознакомиться с двумя структурами Simple MAPI — TMAPIRecipDesc (с ней мы сталкивались в первом примере) и TMAPIFileDesc.

Структура TMAPIRecipDesc представлена в Object Pascal следующим образом:

```
PMapiRecipDesc = ^TMAPIRecipDesc;
MapiRecipDesc = packed record
 ulReserved: Cardinal;
  ulRecipClass: Cardinal;
  lpszName: LPSTR;
  lpszAddress: LPSTR;
  ulEIDSize: Cardinal;
  lpEntryID: Pointer;
 end;
TMAPIRecipDesc = MapiRecipDesc;
```

Она содержит информацию о получателе или отправителе сообщения. Структура имеет следующие поля, представленные в табл. 1.6.

Таблица 1.6. *Поля структуры* TMAPIRecipDesc

Поле	Описание
ulReserved	Поле зарезервировано. Значение поля всегда установлено в ноль
ulRecipClass	Идентификатор типа получателя: MAPI_ORIG — основной получатель сообщения; MAPI_TO — первый получатель сообщения; MAPI_CC — получатель копии сообщения; MAPI_BCC — получатель копии blind copy
lpszName	Указатель на строку, содержащую имя получателя (отправителя)
lpszAddress	Указатель на строку, содержащую E-mail-адрес получателя (отправителя). Значение данного поля очень велико, дело в том, что данное поле, в основном, используется и заполняется системой для входящих сообщений. Для исходящих сообщений программист должен принудительно установить адрес в следующем формате: [тип адреса]:[электронный адрес] где в типе адреса указан протокол отправки, в нашем случае SMTP, например: SMTP:demo@mail.ru
ulEIDSize	Размер в байтах указателя ulEIDSize
lpEntryID	Указатель на идентификатор, используемый Simple MAPI, чтобы установить получателя сообщения

Второй структурой, рассматриваемой нами, является структура TMAPIFileDesc, содержащая информацию о прикрепленных вложениях в сообщении:

```
PMapiFileDesc = ^TMAPIFileDesc;
MapiFileDesc = packed record
 ulReserved: Cardinal;
  flFlags: Cardinal;
  nPosition: Cardinal;
  lpszPathName: LPSTR;
  lpszFileName: LPSTR;
  lpFileType: Pointer;
 end;
TMAPIFileDesc = MapiFileDesc;
```

Структура имеет следующие поля, представленные в табл. 1.7.

Таблица 1.7. *Поля структуры* TMAPIFileDesc

Поле	Описание
ulReserved	Поле зарезервировано. Значение поля всегда установлено в ноль
flFlags	Маска флагов, описывающих тип вложения. В данном параметре могут быть установлены следующие флаги: MAPI_OLE — OLE-объект; MAPI_OLE_STATIC — статичный OLE-объект. Если не установлен ни один флаг, то вложение — простой файл данных
nPosition	Положение файлов-вложений в сообщении. Используется в поле MapiMessage.lpszNoteText
lpszPathName	Указатель на строку, содержащую полный путь к файлу
lpszFileName	Указатель на строку, содержащую имя файла. Данное имя файла увидит получатель сообщения. Может отличаться от имени в lpszPathName, если используется временный файл. Если значение не задано или установлено, как Nil, то используется имя из lpszPathName
lpFileType	Тип файла. Если значение установлено Nil, файл будет воспринят, как не распознанный операционной системой

После небольшого ознакомления модернизируем наше первое приложение. Результат модернизации вы можете наблюдать на рис. 1.2. Полный исходный код расположен на прилагаемом компакт-диске в каталоге Source\Ch01\Ex02.

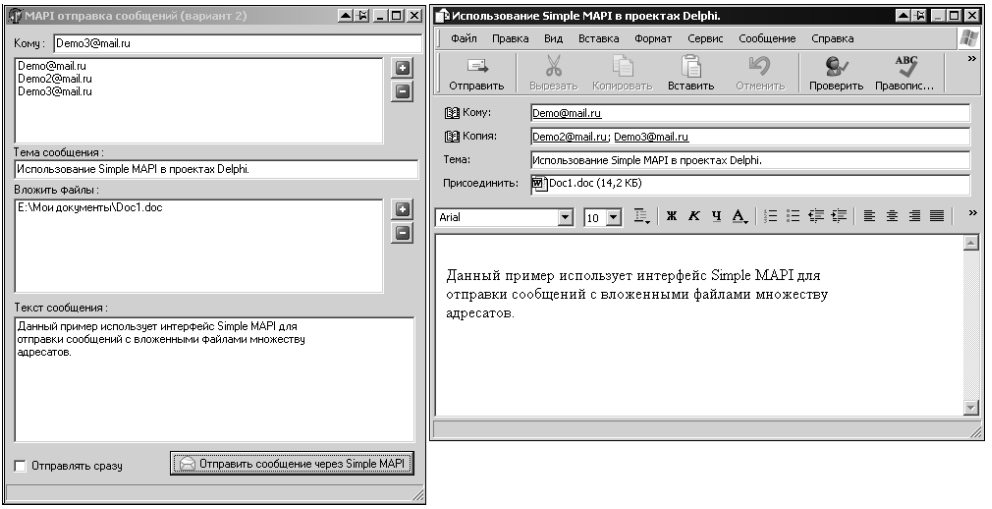

Рис. 1.2. Отправка сообщения

Существенным изменением приложения является то, что в соответствующие поля структуры `TMAPIMessage` передаются не единичные экземпляры соответствующих структур `TMAPIRecipDesc` и `TMAPIFileDesc`, а массивы этих структур, которые и предоставляют возможность множественного добавления получателей и файлов-вложений. Если в первом примере для задания получателя мы передавали просто экземпляр структуры `TMAPIRecipDesc`, то в модернизированном приложении мы передаем массив структур `TMAPIRecipDesc`, выделенных в памяти. Simple MAPI автоматически определяет каждого получателя. Следует также отметить, что в примере при задании множества получателей основным выбирается только один, а все остальные попадают в поле "Копия". Краткий пример создания и работы с массивом получателей представлен в листинге 1.2.

Листинг 1.2. Работа с массивом получателей `TMAPIRecipDesc`

```
type
  // Массив для хранения получателей
  TRecipDescArray = array [0..0] of TMAPIRecipDesc;
  PRecipDescArray = ^TRecipDescArray;
var
  Receip     : PRecipDescArray;    // Указатель на массив получателей.
  ReceipCount: Integer;            // Кол-во получателей
begin
     ReceipCount := ListBoxRecip.Items.Count;
```

```
    if ReceipCount > 0 then
    begin
      // Выделяем область памяти для массива структуры TMAPIRecipDesc
      GetMem(Receip, SizeOf(TMAPIRecipDesc) * ReceipCount);
      For I := 0  to ReceipCount-1 do
      begin
         // Заполняем поле "Получатель"
         RecipName := ListBoxRecip.Items.Strings[I];
         Receip[I].ulReserved   := 0;
         IF I = 0 then
            Receip[I].ulRecipClass := MAPI_TO   // Основной получатель
         else
            Receip[I].ulRecipClass := MAPI_CC;  // Копии

         Receip[I].lpszName     := StrNew(PChar(RecipName));
         Receip[I].lpszAddress  := StrNew(PChar('SMTP:' + RecipName));
         Receip[I].ulEIDSize := 0;
      end;
      MapiMessage.nRecipCount := ReceipCount;
      MapiMessage.lpRecips := @Receip^;
    end

    // Очищаем выделенную память
    // Массивы
    for I := 0 to ReceipCount - 1 do
    begin
        StrDispose(Receip[I].lpszName);
        StrDispose(Receip[I].lpszAddress);
    end;
end;
```

Работа с адресной книгой на Simple MAPI

Для работы с адресной книгой и хранящимися в ней записями в Simple MAPI предусмотрены следующие функции: `MAPIAddress`, `MAPIDetails`, `MAPIResolveName`. Функция `MAPIAddress` создает или модифицирует записи в адресной книге. Данная функция имеет следующий вид:

```
function MapiAddress(lhSession: LHANDLE; ulUIParam: Cardinal;
         lpszCaption: LPSTR; nEditFields: Cardinal;
         lpszLabels: LPSTR;
         nRecips: Cardinal;
         var lpRecips: TMAPIRecipDesc;
```

```
    flFlags: FLAGS;
    ulReserved: Cardinal;
    lpnNewRecips: PULONG;
    var lppNewRecips: PMapiRecipDesc): Cardinal;
```

Параметры функции представлены в табл. 1.8.

Таблица 1.8. *Параметры функции* `MAPIAddress`

Параметр	Описание
`lhSession`	Указатель на открытую ранее сессию Simple MAPI
`ulUIParam`	Дескриптор окна, вызвавшего `MAPIAdress`. Если указан 0, то параметр `ulUIParam` игнорируется
`lpszCaption`	Заголовок окна адресной книги. Если строка пустая, используется название по умолчанию
`nEditFields`	Количество полей редактирования, представленных в адресной книге. Может принимать значения от 0 до 4. При значении поля равном 0 функция предоставляет возможность редактирования записей в адресной книге, остальные значения предоставляют только выбор записей в соответствующие поля "Кому", "Копия" и т. д.
`lpszLabels`	Указатель на строку, используемую в диалоге списка адресов. Если `nEditFields` не равен единице, происходит игнорирование параметра
`nRecips`	Параметр, показывающий, какое число элементов берется из массива `lpRecips`. Если параметр равен нулю, то он игнорируется
`lpRecips`	Указатель на структуру `TMAPIRecipDesc` или на массив структур `TMAPIRecipDesc`. Данный параметр используется для заполнения уже выбранных записей о контактах, которые в свою очередь будут отображаться в соответствующих полях "Кому", "Копия" и т. д., а не в общем списке
`flFlags`	Маска флагов дополнительных опций. В данном параметре могут быть установлены следующие флаги: `MAPI_LOGON_UI` — показывает диалог для идентификации, если это потребуется; `MAPI_NEW_SESSION` — если флаг установлен, функция попытается создать новую сессию, перед тем как использовать существующую
`ulReserved`	Данный параметр зарезервирован
`lpnNewRecips`	Указатель на элемент в массиве `lppNewRecips`. Если параметр равен нулю, то данный параметр игнорируется

Таблица 1.8 *(окончание)*

Параметр	Описание
lppNewRecips	Указатель на структуру TMAPIRecipDesc или на массив структур TMAPIRecipDesc. Данный параметр возвращает записи о контактах, выбранные в адресной книге

Небольшой пример, позволяющий добавлять контакты из адресной книги, представлен в листинге 1.3. Пример максимально упрощен, в частности, мы не стали реализовывать возможность установки уже выбранных контактов в окне адресной книге. Реализацию этого мы оставляем вам для исследования работы функции. Полный исходный текст примера вы сможете найти на прилагаемом компакт-диске в каталоге Source\Ch01\Ex03.

Листинг 1.3. Добавление контактов из адресной книги

```
procedure TFormMain.SpeedButton5Click(Sender: TObject);
var
  dwRet       : Cardinal;
  MAPI_Session: Cardinal;
  I           : Integer;
  lpRecip     : TMAPIRecipDesc;
  intRecips   : ULONG;
  lpRecips    : PMapiRecipDesc;
begin
  // Открываем сессию Simple MAPI
  dwRet := MapiLogon(Application.Handle,PChar(''),PChar(''),
                           MAPI_LOGON_UI or
                           MAPI_NEW_SESSION,0,@MAPI_Session);
  // Если процесс установки почтовой сессии завершился неудачей
  if (dwRet <> SUCCESS_SUCCESS) then
  begin
      ShowMessage('Не могу установить почтовую сессию !');
      SysErrorMessage(dwRet);
  end
  else
  begin
      IF MAPIAddress(MAPI_Session, 0, 'Выбрать пользователей', 1, '', 0,
                     lpRecip,0,0, @intRecips, lpRecips) = SUCCESS_SUCCESS
      Then
      begin
         // Добавляем контакты из адресной книги
         For I := 0 to IntRecips-1 do
```

```
            ListBoxRecip.Items.Add(PMapiRecipDesc(PChar(lpRecips) +
                          I * SizeOf(TMAPIRecipDesc))^.lpszAddress);
        // Очищаем буфер контактов
        MAPIFreeBuffer(lpRecips);
      end;
      // Завершаем сессию
      MapiLogOff(MAPI_Session, Application.Handle, 0, 0);
  end;
end;
```

Функция `MAPIDetails` показывает диалоговое окно, выводящее свойства выбранной записи в адресной книге. Функция имеет вид:

```
function MapiDetails(lhSession: LHANDLE; ulUIParam: Cardinal;
     var lpRecip: TMAPIRecipDesc; flFlags: FLAGS;
     ulReserved: Cardinal): Cardinal;
```

Параметры функции представлены в табл. 1.9.

Таблица 1.9. *Параметры функции* `MAPIDetails`

Параметр	Описание
`lhSession`	Указатель на открытую ранее сессию Simple MAPI
`ulUIParam`	Дескриптор окна, вызвавшего функцию. Если указан 0, то параметр `ulUIParam` игнорируется
`lpRecip`	Указатель на контакт (адресат), для которого надо показать подробности
`flFlags`	Маска флагов дополнительных опций. В данном параметре могут быть установлены следующие флаги: `MAPI_AB_NOMODIFY` — если этот флаг установлен, сведения в диалоговом окне предназначены только для чтения. Пользователь не сможет их изменить; `MAPI_LOGON_UI` — показывает диалог для идентификации, если это потребуется; `MAPI_NEW_SESSION` — если флаг установлен, функция попытается создать новую сессию, перед тем как использовать существующую
`ulReserved`	Данный параметр зарезервирован

Функция `MAPIResolveName` служит для сопоставления имени адресата, внесенного в адресную книгу с электронным адресом, т. е. осуществляет поиск адреса по имени. Функция имеет следующий вид:

```
function MapiResolveName(lhSession: LHANDLE; ulUIParam: Cardinal;
           lpszName: LPSTR;
           flFlags: FLAGS;
           ulReserved: Cardinal;
           var lppRecip: PMapiRecipDesc): Cardinal;
```

Параметры функции представлены в табл. 1.10.

Таблица 1.10. *Параметры функции* `MAPIResolveName`

Параметр	Описание
`lhSession`	Указатель на открытую ранее сессию Simple MAPI
`ulUIParam`	Дескриптор окна, вызвавшего функцию. Если указан 0, то параметр `ulUIParam` игнорируется
`lpszName`	Указатель на строку имени для сопоставления
`flFlags`	Маска флагов дополнительных опций. В данном параметре могут быть установлены следующие флаги: `MAPI_AB_NOMODIFY` — если этот флаг установлен, сведения в диалоговом окне предназначены только для чтения. Пользователь не сможет их изменить; `MAPI_DIALOG` — если указан этот флаг и в адресных книгах найден больше чем один адресат с таким же именем, то будет выведен диалог для выбора имени; `MAPI_LOGON_UI` — показывает диалог для идентификации, если это потребуется; `MAPI_NEW_SESSION` — если флаг установлен, функция попытается создать новую сессию, перед тем как использовать существующую
`ulReserved`	Данный параметр зарезервирован
`lppRecip`	Указатель на структуру `TMAPIRecipDesc`, содержащую сведения об адресате

Помимо основных функций работы с адресной книгой стоит также остановиться на функции `MAPIFreeBuffer`, которую мы использовали в нашем примере.

Данная функция имеет следующий вид:

```
function MapiFreeBuffer(pv: Pointer): Cardinal;
```

Назначение данной функции состоит в том, что она освобождает память, выделенную другими функциями Simple MAPI. В качестве параметра `pv` функ-

ция принимает указатель на буфер памяти. Например, после использования функции MAPIAdress в качестве выбора адресатов система сообщений автоматически выделяет необходимую память для хранения записей в виде массива структур TMAPIRecipDesc. Функция MAPIFreeBuffer призвана корректно освободить занимаемую память во избежание утечки памяти.

Работа с сообщениями на Simple MAPI

Для приема и работы с сообщениями в Simple MAPI реализованы следующие функции:

- MAPIFindNext и MAPIReadMail — для непосредственного приема и чтения сообщений;
- MAPIDeleteMail — для удаления писем;
- MAPISaveMail — для сохранения письма в хранилище сообщений почтового клиента.

Функция MAPIFindNext забирает сообщение, следующее за текущим, и возвращает идентификатор очередного письма из папки "Входящие" (InBox) почтового клиента. Данная функция должна вызываться перед функцией MAPIReadMail, которую мы рассмотрим далее.

Функция MAPIFindNext имеет следующий синтаксис:

```
function MapiFindNext(lhSession: LHANDLE; ulUIParam: Cardinal;
          lpszMessageType: LPSTR;
          lpszSeedMessageID: LPSTR;
          flFlags: FLAGS;
          ulReserved: Cardinal;
          lpszMessageID: LPSTR): Cardinal;
```

Параметры функции MAPIFindNext представлены в табл. 1.11.

Таблица 1.11. *Параметры функции* MAPIFindNext

Параметр	Описание
lhSession	Указатель на открытую ранее сессию Simple MAPI
ulUIParam	Дескриптор окна, вызвавшего функцию. Если указан 0, то параметр ulUIParam игнорируется
lpszMessageType	Указатель на строку-идентификатор класса для поиска
lpszSeedMessageID	Указатель на строку-идентификатор сообщения

Таблица 1.11 *(окончание)*

Параметр	Описание
flFlags	Маска флагов дополнительных опций. В данном параметре могут быть установлены следующие флаги: MAPI_GUARANTEE_FIFO — если флаг установлен, идентификаторы полученных сообщений выводятся в порядке получения (по времени); MAPI_LONG_MSGID — указывает, что идентификатор сообщения превышает 512 символов. При установке данного флага параметр lpszMessageID должен быть больше 512; MAPI_UNREAD_ONLY — указывает на то, что будут выведены только непрочитанные сообщения
ulReserved	Данный параметр зарезервирован
lpszMessageID	Указатель на идентификатор следующего сообщения, т. е. если значение параметра lpszSeedMessageID будет равно 1, то lpszMessageID вернет 2 при условии, что письмо с идентификатором 2 существует

После получения идентификатора сообщения для чтения содержимого нужно вызвать функцию MAPIReadMail, представленную в Simple MAPI в следующем виде:

```
function MapiReadMail(lhSession: LHANDLE; ulUIParam: Cardinal;
         lpszMessageID: LPSTR; flFlags: FLAGS;
         ulReserved: Cardinal;
         var lppMessage: PMapiMessage): Cardinal;
```

Параметры функции представлены в табл. 1.12.

Таблица 1.12. *Параметры функции* `MAPIReadMail`

Параметр	Описание
lhSession	Указатель на открытую ранее сессию Simple MAPI
ulUIParam	Дескриптор окна, вызвавшего функцию. Если указан 0, то параметр ulUIParam игнорируется
lpszSeedMessageID	Указатель на идентификатор сообщения
flFlags	Маска флагов дополнительных опций. В данном параметре могут быть установлены следующие флаги: MAPI_BODY_AS_FILE — показывает, что само сообщение будет сохранено во внешний файл; MAPI_ENVELOPE_ONLY — установка данного флага заставляет Simple MAPI прочитать только заголовок письма;