**8.2 Диалоги открытия и сохранения файлов — компоненты OpenDialog, SaveDialog, OpenPictureDialog, SavePictureDialog**

Компоненты **OpenDialog** — диалог «Открыть файл» и **SaveDialog** — диалог «Сохранить файл как...», пожалуй, используются чаще всего, в большинстве приложений. Примеры открываемых ими диалоговых окон приведены на рис. 8.1 и 8.2.

|  |  |
| --- | --- |
| **Рис. 8.1** Диалоговое окно открытия файла |  |

|  |  |
| --- | --- |
| **Рис. 8.2** Диалоговое окно сохранения файла |  |

Все свойства этих компонентов одинаковы, только их смысл несколько различен для открытия и закрытия файлов. Основное свойство, в котором возвращается в виде строки выбранный пользователем файл, — **FileName**. Значение этого свойства можно задать и перед обращением к диалогу. Тогда оно появится в диалоге как значение по умолчанию в окне Имя файла (см. рис. 8.1, 8.2).

Типы искомых файлов, появляющиеся в диалоге в выпадающем списке Тип файла (рис. 8.1, 8.2), задаются свойством **Filter**. В процессе проектирования это свойство проще всего задать с помощью редактора фильтров, который вызывается нажатием кнопки с многоточием около имени этого свойства в Инспекторе Объектов. При этом открывается окно редактора, вид которого представлен на рис. 8.3. В его левой панели Filter Name вы записываете тот текст, который увидит пользователь в выпадающем списке Тип файла диалога. А в правой панели Filter записываются разделенные точками с запятой шаблоны фильтра. В примере рис. 8.3 задано два фильтра: текстовых файлов с расширениями **.txt** и **.doc** и любых файлов с шаблоном **\*.\***.

|  |  |
| --- | --- |
| **Рис. 8.3** Окно редактора фильтров |  |

После выхода из окна редактирования фильтров заданные вами шаблоны появятся в свойстве **Filter** в виде строки вида:

текстовые (\*.txt, \*.doc)|\*.txt; \*.doc|все файлы|\*.\*

В этой строке тексты и шаблоны разделяются вертикальными линиями. В аналогичном виде, если требуется, можно задавать свойство **Filter** программно во время выполнения приложения.

Свойство **FilterIndex** определяет номер фильтра, который будет по умолчанию показан пользователю в момент открытия диалога. Например, значение **FilterIndex = 1** задает по умолчанию первый фильтр.

Свойство **InitialDir** определяет начальный каталог, который будет открыт в момент начала работы пользователя с диалогом. Если значение этого свойства не задано, то открывается текущий каталог или тот, который был открыт при последнем обращении пользователя к соответствующему диалогу в процессе выполнения данного приложения.

Свойство **DefaultExt** определяет значение расширения файла по умолчанию. Если значение этого свойства не задано, пользователь должен указать в диалоге полное имя файла с расширением. Если же задать значение **DefaultExt**, то пользователь может писать в диалоге имя без расширения. В этом случае будет принято заданное расширение.

Свойство **Title** позволяет вам задать заголовок диалогового окна. Если это свойство не задано, окно открывается с заголовком, определенным в системе (например, «Открытие файла» в окне на рис. 8.1). Но вы можете задать и свой заголовок, подсказывающий пользователю ожидаемые действия. Например, «Укажите имя открываемого файла».

Свойство **Options** определяет условия выбора файла. Множество опций, которые вы можете установить программно или во время проектирования, включает:

|  |  |
| --- | --- |
| **ofAllowMultiSelect** | Позволяет пользователю выбирать несколько файлов |
| **ofCreatePrompt** | В случае, если пользователь написал имя несуществующего файла, появляется замечание и запрос, надо ли создать файл с заданным именем |
| **ofEnableIncludeNotify** | Разрешает посылать в диалог сообщения |
| **ofEnableSizing** | Разрешает пользователю изменять размер диалогового окна |
| **ofExtensionDifferent** | Этот флаг, который можно прочитать после выполнения диалога, показывает, что расширение файла, выбранного пользователем, отличается от **DefaultExt** |
| **ofFileMustExist** | В случае, если пользователь написал имя несуществующего файла, появляется сообщение об ошибке |
| **ofHideReadOnly** | Удаляет из диалога индикатор Открыть только для чтения |
| **ofNoChangeDir** | После щелчка пользователя на кнопке OK восстанавливает текущий каталог, независимо от того, какой каталог был открыт при поиске файла |
| **ofNoDereferenceLinks** | Запрещает переназначать клавиши быстрого доступа в диалоговом окне |
| **ofNoLongNames** | Отображаются только не более 8 символов имени и трех символов расширения |
| **ofNoNetworkButton** | Убирает из диалогового окна кнопку поиска в сети. Действует только если флаг **ofOldStyleDialog** включен |
| **ofNoReadOnlyReturn** | Если пользователь выбрал файл только для чтения, то генерируется сообщение об ошибке |
| **ofNoTestFileCreate** | Запрещает выбор в сети защищенных файлов и не доступных дисков при сохранении файла |
| **ofNoValidate** | Не позволяет писать в именах файлов неразрешенные символы, но не мешает выбирать файлы с неразрешенными символами |
| **ofOldStyleDialog** | Создает диалог выбора файла в старом стиле (см. рис. 8.4) |
| **ofOverwritePrompt** | В случае, если при сохранении файла пользователь написал имя существующего файла, появляется замечание, что файл с таким именем существует, и запрашивается желание пользователя переписать существующий файл |
| **ofPathMustExist** | Генерирует сообщение об ошибке, если пользователь указал в имени файла несуществующий каталог |
| **ofReadOnly** | По умолчанию устанавливает индикатор Открыть только для чтения при открытии диалога |
| **ofShareAware** | Игнорирует ошибки нарушения условий коллективного доступа и разрешает, несмотря на них, производить выбор файла |
| **ofShowHelp** | Отображает в диалоговом окне кнопку Справка |

По умолчанию все перечисленные опции, кроме **ofHideReadOnly**, выключены. Но, как видно из их описания, многие из них полезно включить перед вызовом диалогов.

Если вы разрешаете с помощью опции **ofAllowMultiSelect** множественный выбор файлов, то список выбранных файлов можно прочитать в свойстве **Files** типа **TStrings**.

В приведенной таблице даны опции, используемые в 32-разрядных версиях Delphi. В Delphi 1 диалоговое окно имеет вид, представленный на рис. 8.4. Аналогичный вид имеет диалог и в 32-разрядных версиях Delphi при включении опции **ofOldStyleDialog**. В примере рис. 8.4 диалог открыт с заданным значением свойства **Title** и заданный текст отображается в заголовке окна. Кроме того, в этом примере выключена опция **ofHideReadOnly** (в Delphi 1 она выключена по умолчанию), что привело к появлению индикатора «Только чтение».

|  |  |
| --- | --- |
| **Рис. 8.4** Диалог в старом стиле при включенной опции **ofOldStyleDialog** и выключенной опции **ofHideReadOnly** |  |

В компонентах диалогов открытия и сохранения файлов предусмотрена возможность обработки ряда событий. Такая обработка может потребоваться, если рассмотренных опций, несмотря на их количество, не хватает, чтобы установить все диктуемые конкретным приложением ограничения на выбор файлов. Событие **OnCanClose** возникает при нормальном закрытии пользователем диалогового окна после выбора файла. При отказе пользователя от диалога — нажатии кнопки Отмена, клавиши Esc и т.д. событие **OnCanClose** не наступает. В обработке события **OnCanClose** вы можете произвести дополнительные проверки выбранного пользователем файла и, если по условиям вашей задачи этот выбор недопустим, вы можете известить об этом пользователя и задать значение **false** передаваемому в обработчик параметру **CanClose**. Это не позволит пользователю закрыть диалоговое окно.

Можно также написать обработчики событий **OnFolderChange** — изменение каталога, **OnSelectionChange** — изменение имени файла, **OnTypeChange** — изменение типа файла. В этих обработчиках вы можете предусмотреть какие-то сообщения пользователю.

Теперь приведем примеры использования диалогов **OpenDialog** и **SaveDialog**. Пусть ваше приложение включает окно редактирования **Memo1**, в которое по команде меню Открыть вы хотите загружать текстовый файл, а после каких-то изменений, сделанных пользователем, — сохранять по команде Сохранить текст в том же файле, а по команде Сохранить как... — в файле с другим именем.

Введите на форму компоненты — диалоги **OpenDialog** и **SaveDialog**. Предположим, что вы оставили их имена по умолчанию — **OpenDialog1** и **SaveDialog1**. Поскольку после чтения файла вам надо запомнить его имя, чтобы знать под каким именем потом его сохранять, вы можете определить для этого имени переменную, назвав ее, например, **FName**:

**var** FName: **string**;

Тогда обработка команды Открыть может сводиться к следующему оператору:

**if** OpenDialog1.Execute **then**

**begin**

FName := OpenDialog1.FileName;

Memo1.Lines.LoadFromFile(FName);

**end**;

Этот оператор вызывает диалог, проверяет, выбрал ли пользователь файл (если выбрал, то функция **Execute** возвращает **true**), после чего имя выбранного файла (**OpenDialog1.FileName**) сохраняется в переменной **FName** и файл загружается в текст **Memo1** методом **LoadFromFile**.

Обработка команды Сохранить выполняется оператором

Memo1.Lines.SaveToFile(FName);

В данном случае нет необходимости обращаться к какому-то диалогу, поскольку имя файла известно: оно хранится в переменной **FName**.

Обработка команды Сохранить как... выполняется операторами:

SaveDialog1.FileName := FName;

**if** SaveDialog1.Execute **then**

**begin**

FName := SaveDialog1.FileName;

Memo1.Lines.SaveToFile(FName);

**end**;

Первый из этих операторов присваивает свойству **FileName** компонента **SaveDialog1** запомненное имя файла. Это имя по умолчанию будет предложено пользователю при открытии диалога Сохранить как.... Следующий оператор открывает диалог и, если пользователь выбрал в нем файл, запоминает новое имя файла и сохраняет в файле с этим именем текст компонента **Memo1**.

Мы рассмотрели диалоги открытия и сохранения файлов произвольного типа. Начиная с Delphi 3 в библиотеке имеются специализированные диалоги открытия и закрытия графических файлов: **OpenPictureDialog** и **SavePictureDialog**. Диалоговые окна, открываемые этими файлами, приведены на рис. 8.5 и 8.6. От окон, открываемых компонентами **OpenDialog** и **SaveDialog** (рис. 8.1, 8.2), они отличаются удобной возможностью просматривать изображения в процессе выбора файла.

|  |  |
| --- | --- |
| **Рис. 8.5** Диалоговое окно открытия файла изображения |  |
|  | |

|  |  |
| --- | --- |
| **Рис. 8.6** Диалоговое окно сохранения файла изображения |  |
|  | |

Свойства компонентов **OpenPictureDialog** и **SavePictureDialog** ничем не отличаются от свойств компонентов **OpenDialog** и **SaveDialog**. Единственное отличие — заданное значение по умолчанию свойства **Filter** в **OpenPictureDialog** и **SavePictureDialog**. В этих компонентах заданы следующие фильтры:

|  |  |
| --- | --- |
| All (\*.jpg; \*.jpeg; \*.bmp; \*.ico; \*.emf; \*.wmf) | \*.jpg; \*.jpeg; \*.bmp; \*.ico; \*.emf; \*.wmf |
| JPEG Image File (\*.jpg) | \*.jpg |
| JPEG Image File (\*.jpeg) | \*.jpeg |
| Bitmaps (\*.bmp) | \*.bmp |
| Icons (\*.ico) | \*.ico |
| Enhanced Metafiles (\*.emf) | \*.emf |
| Metafiles (\*.wmf) | \*.wmf |

В этих фильтрах перечислены все типы графических файлов, с которыми может работать диалог. Так что вам остается удалить, если хотите, фильтры тех файлов, с которыми вы не хотите работать, добавить, может быть, фильтр «Все файлы (\*.\*)» и перевести на русский язык названия типов.