4

The EiffelCOM Library

The EiffelCOM library adds compound files support to your Eiffel applications. Compound files are structured files that can embed different types of data in a single file. It is the ideal format to store documents including different types of information. It is the format used by Microsoft® Office applications. This chapter will not cover the compound files architecture itself but will focus on its support in EiffelCOM. It requires that the user be familiar with compound files and the compound files API.

The EiffelCOM library enables the creation and use of the **IRootStorage**, **IStorage** and **IStream** interfaces and includes a wrapping of all necessary structures for handling compound files. It also includes classes that encapsulates all necessary flags and constants.

4.1 COMPOUND FILES

The cluster **ecom_storage** includes the files *ecom_root_storage.e*, *ecom_storage.e* and *ecom_stream.e*. These files correspond to the **IRootStorage, IStorage** and **IStream** interfaces, respectively.

Storages

Storages are to compound files what directories are to a standard file system. They can include nested storages and/or streams. Streams are the equivalent of files in a standard file system. They include the data itself. The following features are available on *ECOM_STORAGE* objects:

- *destroy_element* (*element_name*: **STRING**) — removes the stream or the substorage *element_name* of **Current**. You can cancel this action using a call to *revert* (described later). Thus, *commit* (described later) will be called to confirm the deletion.
- *copy_to* (*stg_dest*: *ECOM_STORAGE*) — copies **Current** into *stg_dest*.
- *move_element_to* (*element_name*: *STRING*; *stg_dest*: *ECOM_STORAGE*; *new_element_name*: *STRING*; *movmode*: *INTEGER*) — moves the stream or substorage element_name to *stg_dest* and renames *element_name* as *new_element_name*. *movmode* can take one of the values described in **ECOM_STGMOVE**.

- *rename_element* (*old_element_name*, *new_element_name*: *STRING*) — renames the stream or substorage *old_elment_name* as *new_element_name*.
- *set_class* (*clsid*: *STRING*) — set the class identifier of **Current** using the *clsid* value.
- *commit* (*mode*: *INTEGER*) — commits all changes made in **Current**. The commits all streams and substorages included in **Current**. *mode* can take one of the values described in **ECOM_STGC**.
- *revert* — discards all changes made to **Current**.
- *is_compound_file* (*filename*: *STRING*): *BOOLEAN* — checks if the system file *filename* is a compound file (**Result** = **True**) or not (**Result** = **False**).
- *enum_elements*: *ARRAY* [*ECOM_STATSTG*] — enumerates the **ECOM_STATSTG** structures associated with the streams and substorages included in **Current**.

Streams

Streams are where the data is actually stored. Different streams can store different types of data. The following features are available on *ECOM_STREAM* objects:

- *seek* (*offset*, *origin*: *INTEGER*) — sets the position of the seek pointer by adding *offset* to *origin*. *origin* can contain one of the values described in **ECOM_STREAM_SEEK**.
- *set_size* (*new_size*: *INTEGER*) — sets the size of the stream to *new_size*. If the *new_size* value is larger than the current size of the stream, then the new bytes fill with undefined values. If the *new_size* value is smaller than the current size, the stream truncates.
- *copy_to* (*stream_dest*: *ECOM_STREAM*; *num_bytes_to_copy*: *INTEGER*) — copies *num_bytes_to_copy* to *stream_dest*.
- *commit* (*mode*: *INTEGER*) — commits all changes made to the stream. *mode* can take one of the values described in **ECOM_STGC**. Compound file implementation does not support opening streams in transacted mode, so this method primarily flushes memory buffers.
- *revert* — discards all changes made to the stream. Compound file implementation does not support opening streams in transacted mode, so this method has no effect.
- *stat* (*stat_flag*: *INTEGER*) — fills an **ECOM_STATSTG** structure that corresponds to current stream.
- *read* (*buff*: *POINTER*; *num_bytes_to_read*: *INTEGER*): *INTEGER* — reads *num_bytes_to_read* bytes from the stream into *buff*. Returns the actual number of bytes read.
- *write* (*buff*: *POINTER*; *num_bytes_to_write*: *INTEGER*): *INTEGER* — writes the *num_bytes_to_write* bytes of buffer pointed by *buff* to stream, and the returns the number of bytes actually written.

- *lock_region* (*offset*, *count*: *ECOM_LARGE_INTEGER*; *lock*: *INTEGER*) — restricts access to the range of bytes defined by *offset* and *count*. *lock* can take one of the values described in **ECOM_LOCKTYPES**.
- *unlock_region* (*offset*, *count*: *ECOM_LARGE_INTEGER*; *lock*: *INTEGER*) — removes the access restriction previously set using *lock_region*. The arguments are the same as for *lock_region*.

The last two features may of may not be available, depending on the COM interface that the Eiffel class wraps. If the interface does not support region locking, then the status code is set to *Stg_e_invalidfunction*. The standard Windows implementation of compound file does not support region locking.

All of the previous features directly encapsulate the COM **IStream** interface. Thus, read/write is not very effective, since it requires a pointer on a buffer, which is not directly available in Eiffel. As a result, the **ECOM_STREAM** class offers the following read/write features:

- *read_string* — reads a string at the current seek position in the stream and sets *last_string* accordingly.
- *read_character*— reads a character at the current seek position in the stream and sets *last_character* accordingly.
- *read_integer*— reads an integer at the current seek position in the stream and sets *last_integer* accordingly.
- *read_real*— reads a real at the current seek position in the stream and sets *last_real* accordingly.
- *read_boolean*— reads a Boolean at the current seek position in the stream and sets *last_boolean* accordingly.
- *write_string* (*s*: *STRING*) — writes *s* at the current seek position in the stream.
- *write_character* (*c*: *CHARACTER*) — writes *c* at the current seek position in the stream.
- *write_integer* (*i*: *INTEGER*) — writes *i* an integer at the current seek position in the stream.
- *write_real* (*r*: *REAL*) — writes *r* at the current seek position in the stream.
- *write_boolean* (*b*: *BOOLEAN*) — writes *b* at the current seek position in the stream.
- *end_of_stream*: *BOOLEAN* — **True** when the end of the stream is reached.

The attribute *end_of_stream* automatically updates when there is a call to any of the read/write features.

Other classes

The EiffelCOM compound file interfaces also need several other classes. These classes handle the **ECOM_STATSTG** structure included in the *ecom_structure* cluster and require some of the constants defined in the *ecom_flags* cluster.

The **ECOM_STATSTG** class is a direct encapsulation of the **STATSTG** structure. The following attributes can be set/accessed

- *element_name*: *STRING* — name of element (storage or stream).
- *element_type*: *INTEGER* — type of element (storage, stream). **Result** can take one of the values described in **ECOM_STGTY**.
- *element_size*: *INTEGER* — size of element in bytes (stream).
- *modification_time*: *WEL_FILE_TIME* — last modification time.
- *creation_time*: *WEL_FILE_TIME* — creation time.
- *access_time*: *WEL_FILE_TIME* — last access time.
- *open_mode*: *INTEGER* — mode in which element was opened. **Result** can take one of the values described in **ECOM_STGM**.
- *locks_supported*: *INTEGER* — A group of flags relevant only for stream. For each lock item, indicates whether or not a call to *lock_region* is worthwhile. **Result** can take one of the values described in **ECOM_LOCK_TYPE**.
- *clsid*: *STRING* — class identifier associated with the storage.
- *state_bits*: *INTEGER* — last state bits set on element (storage).

The following are the classes that encapsulate the COM constants:

- **ECOM_LOCK_TYPES —** defines the different lock types available for a storage element.
- **ECOM_STAT_FLAGS —** specifies which field of the **ECOM_STATSTG** structure associated with the element to fill.
- **ECOM_STGC —** SToraGe Commit mode; defines the various available commit modes.
- **ECOM_STGM —** SToraGe Mode; defines the opening mode for an element (storage or stream).
- **ECOM_STGMOVE —** SToraGe MOVE; defines how to move a storage.
- **ECOM_STGTY —** SToraGe TYpe; defines the element type: storage, stream or lockbyte.
- **ECOM_STREAM_SEEK** — defines the current position of the seek pointer in a stream.

Summary

The EiffelCOM library gives access to the main compound files functionality. It allows to create, edit and delete compound files. If your application needs to save different types of data in one single file then compound files provide an easy and straightforward support.