**Функция SetFilePointer**

Функция **SetFilePointer** перемещает указатель позиции в открытом файле.

Эта функция хранит указатель позиции в файле в двух значениях **LONG**. Чтобы облегчить работу с указателями позиции в файле, которые являются большими, чем одиночное значение **LONG**, используйте функцию **[SetFilePointerEx](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_setfilepointerex.htm)**.

**Синтаксис**

|  |
| --- |
| **DWORD SetFilePointer(**  **HANDLE** *hFile***,** // дескриптор файла  **LONG** *lDistanceToMove***,** // байты перемещения указателя  **PLONG** *lpDistanceToMoveHigh***,**  // байты перемещения указателя  **DWORD** *dwMoveMethod* // точка отсчета  **);** |

**Параметры**

***hFile***

[in] Дескриптор файла, указатель позиции в котором должен переместиться. Дескриптор файла, должен быть создан с правом доступа **GENERIC\_READ** или **GENERIC\_WRITE**. Для получения дополнительной информации, см. статью [**Защита файла и права доступа**](http://vsokovikov.narod.ru/New_MSDN_API/File_system/file_access_right.htm).

***lDistanceToMove***

[in] Младшие 32 бита значения со знаком, устанавливающее число байтов, на которое переместится указатель позиции в файле. Если параметр ***lpDistanceToMoveHigh*** - не ПУСТО (NULL), параметры ***lpDistanceToMoveHigh*** и ***lDistanceToMove*** формируют единое 64-разрядное значение со знаком, которое определяет расстояние продвижения. Если ***lpDistanceToMoveHigh*** - ПУСТО (NULL), ***lDistanceToMove*** - 32-разрядное значение со знаком. Положительное значение параметра ***lDistanceToMove*** продвигает указатель позиции в файле вперед, а отрицательное значение перемещает указатель позиции в файле назад.

***lpDistanceToMoveHigh***

[in] Указатель на старшие 32 бита 64-разрядного расстояния со знаком, на которое продвигается указатель позиции. Если Вам не нужны эти старшие 32 бита, этот указатель должен быть установлен в ПУСТО (NULL). Если он равен не ПУСТО (NULL), этот параметр также получает и старшее двойное слово (**DWORD**) нового значения указателя позиции в файле. Для получения дополнительной информации, см. раздел Замечаний ниже в этой статье.

**Windows 95/98/Me:** Если указатель ***lpDistanceToMoveHigh*** - не ПУСТО (NULL), то он должен указать или на 0, **INVALID\_SET\_FILE\_POINTER**, или на значение дополнительного знакового разряда ***lDistanceToMove***. Любое другое значение должно быть отклонено.

***dwMoveMethod***

[in] Точка отсчета для перемещения указателя позиции в файле. Этим параметром может быть одно из нижеперечисленных значений.

|  |  |
| --- | --- |
| **Значение** | **Предназначение** |
| **FILE\_BEGIN** | Точка отсчета равняется нулю или это начало файла. |
| **FILE\_CURRENT** | Точка отсчета - текущее значение указателя позиции в файле. |
| **FILE\_END** | Точка отсчета - текущая позиция метки конца файла. |

**Возвращаемые значения**

Если функция **SetFilePointer** завершается успешно, а параметр ***lpDistanceToMoveHigh*** - ПУСТО (NULL), возвращаемое значение - младшее двойное слово (**DWORD**) нового указателя позиции в файле. Если ***lpDistanceToMoveHigh*** - не ПУСТО (NULL), функция возвращает младшее двойное слово (**DWORD**) нового указателя позиции в файле, и помещает старшее двойное слово (**DWORD**) нового указателя позиции в файле в указателе типа **LONG**  в качестве этого параметра.

Если функция завершается ошибкой, а параметр ***lpDistanceToMoveHigh*** - ПУСТО (NULL), возвращаемое значение - **INVALID\_SET\_FILE\_POINTER**. Чтобы получить дополнительные данные об ошибке, вызовите **[GetLastError](http://vsokovikov.narod.ru/New_MSDN_API/Debbag_error/fn_getlasterror.htm)**.

Если функция завершается ошибкой, а параметр ***lpDistanceToMoveHigh*** - не ПУСТО (NULL), возвращаемое значение - **INVALID\_SET\_FILE\_POINTER**. Однако, поскольку **INVALID\_SET\_FILE\_POINTER** - допустимое значение для младшего двойного слова (**DWORD**) нового указателя позиции в файле, Вы должны проверить при помощи **GetLastError**, чтобы выяснить, не произошла ли ошибка. Если ошибка произошла, **GetLastError** возвращает значение другое, а не **NO\_ERROR**. Пример кода, который иллюстрирует этот момент, смотрите в разделе Замечаний ниже в этой статье.

Если новый указатель позиции в файле был бы отрицательным значением, функция завершается ошибкой, указатель позиции в файле не перемещается, а код, возвращенный **GetLastError** - **ERROR\_NEGATIVE\_SEEK**.

**Замечания**

Указатель позиции в файле, идентифицированный значением параметра ***hFile***  не используется для асинхронных операций чтения и записи. Чтобы определить смещение для асинхронных операций, используйте члены ***Offset*** и ***OffsetHigh*** структуры [**OVERLAPPED**](http://vsokovikov.narod.ru/New_MSDN_API/Synchronization/str_overlapped.htm).

Вы не можете использовать функцию **SetFilePointer** с дескриптором не способного к поиску устройства, такого как канал или коммуникационное устройство. Чтобы выяснить тип файла для ***hFile***, используйте функцию **[GetFileType](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_getfiletype.htm)**.

Чтобы выявить существующее местоположение указателя позиции в файле, см. статью [**Установка положения указателя позиции в файле**](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/position_pointer.htm).

Проявляйте осмотрительность при установке указателя позиции в файле в многопоточном  приложении. Вы должны синхронизировать доступ к совместно используемым ресурсам. Например, прикладная программа, потоки которой совместно используют дескриптор файла, модернизирует указатель позиции в файле и читая из него, она должна защищать эту последовательность байтов, используя объект критической секции или объект типа мьютекс. Для получения дополнительной информации об этих объектах, см. статьи Объекты критической секции и Объекты типа мьютекс.

Если ***hFile*** дескриптор файла открывался с установкой флажка **FILE\_FLAG\_NO\_BUFFERING**, приложение может переместить указатель позиции в файле только в выровненное по границе сектора положение. Выровненное по границам сектора положение - это позиция, которая является целым числом, кратным размеру сектора тома. Приложение может получить размер сектора тома при помощи вызова функции **GetDiskFreeSpace**. Если приложение вызывает функцию **SetFilePointer** со значениями длины перемещения, которые дают в результате позицию, не выровненную по границе сектора, а дескриптор, который открывался, с флажком **FILE\_FLAG\_NO\_BUFFERING**, функция завершается ошибкой, а **GetLastError** возвращает значение **ERROR\_INVALID\_PARAMETER**.

|  |
| --- |
| **Обратите внимание!** на то, что это - не ошибка установки указателя позиции в файле в положение вне конца файла. Размер файла не увеличиться до тех пор, пока Вы не вызовите функцию **[SetEndOfFile](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_setendoffile.htm)**, **[WriteFile](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_writefile.htm)** или **[WriteFileEx](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_writefileex.htm)**. Операция записи увеличивает размер файла до  позиции указателя в файле плюс размер  буфера записи, оставляя промежуточные байты неинициализированными. |

Если возвращаемое значение - **INVALID\_SET\_FILE\_POINTER** и, если ***lpDistanceToMoveHigh*** - не ПУСТО (NULL), приложение должно вызвать **GetLastError**, чтобы выяснить, успешно ли отработала функция или завершилась ошибкой. Нижеследующий типовой код иллюстрирует этот момент:

|  |
| --- |
| // Случай первый: вызов функция с lpDistanceToMoveHigh == NULL    // Попробуем переместить указатель позиции в файле,  // определенном в параметре hFile, на некоторое расстояние.  dwPtr = **SetFilePointer** (hFile, lDistance, NULL, FILE\_BEGIN) ;    **if** (dwPtr **==** INVALID\_SET\_FILE\_POINTER) // Проверка на ошибку  **{**  // Получим код ошибки.  dwError = **GetLastError**() ;    // Имеем дело со сбоем.  // . . .    **}** // Конец обработчика ошибки  //  // Случай второй: вызов функции с lpDistanceToMoveHigh != NULL    // Поробуем переместить указатель позиции в файле,  // определенном в параметре hFile, на некоторое  // очень большое расстояние.  dwPtrLow = **SetFilePointer** (hFile, lDistLow, **&** lDistHigh, FILE\_BEGIN) ;    // Проверка на ошибку  **if** (dwPtrLow **==** INVALID\_SET\_FILE\_POINTER **&&** (dwError = **GetLastError**()) != NO\_ERROR )  **{**  // Имеем дело со сбоем.  // . . .  **}** // Конец обработчика ошибки |

Хотя параметр ***lpDistanceToMoveHigh*** используется, чтобы управлять огромными файлами, значение этого параметра должно устанавливаться при перемещении в файлах любого размера. Если он устанавливается в ПУСТО (NULL), то ***lDistanceToMove*** имеет максимальное значение 231-2, или 2 гигабайта без двух байтов. Это так потому, что все значения указателя позиции в файле - значения со знаком. Поэтому, даже если есть  маленький шанс, что файл будет расти до этого размера, вам следует рассматривать его как огромный файл и работать с 64-разрядными указателями позиции в файле. При помощи [**сжатия файла**](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fl_compres_decompres.htm) (file compression) в **NTFS**, и [**разреженных файлов**](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/sparsel_disk_quota.htm) (sparse files), возможно иметь файлы, которые являются большими, даже если базовый том не является очень большим.

Если ***lpDistanceToMoveHigh*** - не ПУСТО (NULL), то ***lpDistanceToMoveHigh*** и ***lDistanceToMove*** формируют единое 64-разрядное значение со знаком. Параметр ***lDistanceToMove***  рассматривают как младшие 32 бита значения, а ***lpDistanceToMoveHigh*** как старшие 32 бита. В соответствии с этим, ***lpDistanceToMoveHigh*** - дополнительный знаковый разряд ***lDistanceToMove***.

Чтобы переместить указатель позиции в файле от нуля до 2 гигабайтов, ***lpDistanceToMoveHigh*** должен быть установлен  или в ПУСТО (NULL) или в значение дополнительного знакового разряда ***lDistanceToMove***. Чтобы переместить указатель больше чем на 2 гигабайта, используйте ***lpDistanceToMoveHigh*** и ***lDistanceToMove*** как единую 64-разрядную величину. Например, чтобы передвинуться в диапазон от 2  до 4 гигабайтов, установите содержимое ***lpDistanceToMoveHigh*** в нуль, или в -(минус)1 для отрицательного дополнительного знакового разряда ***lDistanceToMove***.

Чтобы работать с 64-разрядными указателями позиции в файле, Вы можете объявить его типом **LONG,** рассматривая его как старшую половину 64-разрядного указателя позиции в файле и передавая его адрес в параметр ***lpDistanceToMoveHigh***. Это означает, что Вы должны рассматривать две различные переменные как один логический модуль, который  подвержен ошибкам. Проблемы могут быть смягчены при помощи использования структуры **LARGE\_INTEGER**, чтобы создать 64-разрядное значение и передавать два 32-разрядных значения посредством соответствующих элементов союза.

Есть концептуально более простой и лучший способ использования функции, чтобы скрыть интерфейс  **SetFilePointer**. Чтобы сделать это, используйте кое-что подобное ниже написанному:

|  |
| --- |
| **\_\_int64** **myFileSeek** (**HANDLE** hf, **\_\_int64** distance, **DWORD** **MoveMethod**)  **{**  **LARGE\_INTEGER** li;  li.QuadPart = distance;  li.LowPart = **SetFilePointer** (hf, li.LowPart, **&**li.HighPart,  **MoveMethod**);  **if** (li.LowPart **==** INVALID\_SET\_FILE\_POINTER **&&** GetLastError()  **!=** NO\_ERROR)  **{**  li.QuadPart = -1;  **}**  **return** li.QuadPart;  **}** |

|  |
| --- |
| **Обратите внимание!**на то, что Вы можете использовать **SetFilePointer**, чтобы выяснить длину файла. Чтобы сделать это, используйте флажок **FILE\_END** для ***dwMoveMethod*** и ищите местоположение нуля. Смещение файла возвращает его длину. Однако, эта практика может иметь непреднамеренные побочные эффекты, типа сбоя, чтобы сохранить текущий указатель позиции в файле так, чтобы программа могла возвратиться этому месту. Проще и безопаснее вместо этого использовать функцию **[GetFileSize](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_getfilesize.htm)**. |

Вы можете также использовать функцию **SetFilePointer**, чтобы сделать запрос текущего местоположения указателя позиции в файле. Чтобы сделать это, определите метод перемещения **FILE\_CURRENT** и нулевое расстояние.

**Код примера**

Пример смотри в статье [**Добавление одного файла  в конец  другого файла**](http://vsokovikov.narod.ru/New_MSDN_API/Menage_files/append_file_another.htm).