<http://www.vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_readfile.htm>

ReadFile

читает данные из файла, начиная с позиции, обозначенной указателем файла. После того, как операция чтения была закончена, указатель файла перемещается на число действительно прочитанных байтов, если дескриптор файла не создан с атрибутом асинхронной операции. Если дескриптор файла создается для асинхронного ввода - вывода, приложение должно переместить позицию указателя файла после операции чтения.

Эта функция предназначена и для синхронной и асинхронной операции. Функция [ReadFileEx](http://www.vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_readfileex.htm) предназначена исключительно для асинхронной операции. Это дает возможность приложению выполнять другие действия в ходе операции чтения файла.

Синтаксис

|  |
| --- |
| BOOL ReadFile(  HANDLE hFile, // дескриптор файла  LPVOID lpBuffer, // буфер данных  DWORD nNumberOfBytesToRead, // число байтов для чтения  LPDWORD lpNumberOfBytesRead, // число прочитанных байтов  LPOVERLAPPED lpOverlapped // асинхронный буфер  ); |

Параметры

hFile

[in] Дескриптор файла, который читается. Дескриптор файла должен быть, создан с правом доступа GENERIC\_READ. Подробную информацию см.  в статье [Защита файла и права доступа](http://www.vsokovikov.narod.ru/New_MSDN_API/File_system/file_access_right.htm).

Windows NT/2000/XP: Для асинхронных операций чтения, параметр hFile может быть любым дескриптором открытым с флажкомFILE\_FLAG\_OVERLAPPED функцией [CreateFile](http://www.vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_createfile.htm) или дескриптор сокета, возвращенный функцией socket или accept.

Windows 95/98/Me: Для асинхронных операций чтения, параметр hFile может быть  коммуникационным ресурсом, открытым с флажком FILE\_FLAG\_OVERLAPPED функцией CreateFile или дескриптор сокета, возвращенный при помощи функций socket илиaccept . Вы не можете выполнять асинхронные операции чтения в почтовом ящике в ядре Windows, именованных каналах или файлах на диске.

lpBuffer

[out] Указатель на буфер, который принимает прочитанные данные из файла.

nNumberOfBytesToRead

[in] Число байтов, которые читаются из файла.

lpNumberOfBytesRead

[out] Указатель на переменную, которая получает число прочитанных байтов. Функция ReadFile устанавливает это значение в нуль перед началом любой работы или проверкой ошибок. Если этот параметр равняется нулю, когда ReadFile возвращает значение ИСТИНА (TRUE) для именованного канала, другой конец канала в режиме передачи сообщений вызывает функцию [WriteFile](http://www.vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_writefile.htm) с параметром nNumberOfBytesToWrite  установленным в нуль.

Windows NT/2000/XP: Если параметр lpOverlapped имеет значение ПУСТО (NULL), параметр lpNumberOfBytesRead не может быть значением ПУСТО (NULL). Если lpOverlapped имеет -  значение не ПУСТО (NULL), lpNumberOfBytesRead может быть значением ПУСТО (NULL). Если это - асинхронная операция чтения, Вы можете получить число прочитанных байтов при помощи вызова функции GetOverlappedResult. Если параметр hFile связан с портом завершения ввода-вывода, Вы можете получить число прочитанных байтов при помощи вызова функции [GetQueuedCompletionStatus](http://www.vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_getqueuedcompletionstatus.htm).

Если используются порты завершения ввода-вывода, а Вы используете процедуру повторного вызова, чтобы освободить занимаемую память структурой [OVERLAPPED](http://www.vsokovikov.narod.ru/New_MSDN_API/Synchronization/str_overlapped.htm), на которую указывает параметр lpOverlapped, задайте  ПУСТО (NULL), как значение этого параметра, чтобы избежать проблемы искажения данных в памяти в ходе ее освобождения. Эта проблема искажения данных в памяти становится причиной возвращения в этом параметре неверного числа байтов .

Windows 95/98/Me: Этот параметр не может быть ПУСТО (NULL).

lpOverlapped

[in] Указатель на структуру OVERLAPPED. Эта структура требуется тогда, если параметр hFile создавался с флажкомFILE\_FLAG\_OVERLAPPED.

Если hFile был открыт с флажком FILE\_FLAG\_OVERLAPPED, у параметра lpOverlapped  не должно быть значения ПУСТО (NULL). Он должен указать на правильную структуру OVERLAPPED. Если hFile создавался с флажком FILE\_FLAG\_OVERLAPPED, аlpOverlapped имеет значение ПУСТО (NULL), функция может неправильно сообщить о завершении операций чтения.

Если hFile был открыт с флажком FILE\_FLAG\_OVERLAPPED, а lpOverlapped имеет значение не ПУСТО (NULL), операция чтения начинается при смещении, заданном в структуре OVERLAPPED, и ReadFile может возвратить значение прежде, чем операция чтения будет закончена. В этом случае, ReadFile возвращает значение ЛОЖЬ (FALSE), а функция [GetLastError](http://www.vsokovikov.narod.ru/New_MSDN_API/Debbag_error/fn_getlasterror.htm) возвращает значениеERROR\_IO\_PENDING. Это дает возможность вызывающему процессу продолжиться, в то время как операция чтения заканчивается. Событие, определяемое в структуре OVERLAPPED устанавливается в сигнальное состояние после завершения операции чтения. Вызывающая программа должна корректировать местоположение указателя позиции в файле после завершения работы.

Функция ReadFile сбрасывает событие, указанное членом hEvent структуры OVERLAPPED в несигнальное состояние, когда она начинает операцию ввода-вывода (I/O). Поэтому, нет необходимости для вызывающей программы, чтобы делать так.

Если hFile не открывался с флажком  FILE\_FLAG\_OVERLAPPED, а lpOverlapped - значение ПУСТО (NULL), операции чтения начинается в текущей позиции файла и ReadFile не возвращает значения до тех пор, пока операция не будет закончена. Система модернизирует указатель позиции в файле после завершения работы.

Если hFile не открывался с FILE\_FLAG\_OVERLAPPED, а lpOverlapped - не ПУСТО (NULL),  операция чтения стартует при смещении, указанном в структуре OVERLAPPED. ReadFile не возвращает значение до тех пор, пока операция чтения не завершилась. Система модернизирует позицию указателя  в файле после завершения работы.

Windows 95/98/Me: Для операций с файлами, дисками, каналами или почтовыми ящиками в ядре Windows, этим параметром должно быть значение ПУСТО (NULL); указатель на структуру OVERLAPPED порождает вызов, который завершиться ошибкой. Тем не менее, Windows 95/98/Me поддерживает асинхронные операции на последовательных и параллельных портах.

Возвращаемые значения

Функция ReadFile возвращает значение тогда, когда выполнено одно из ниже перечисленных  условий:

- операция записи завершается на записывающем конце канала,   
- затребованное число байтов прочитано,   
- или происходит ошибка.

Если функция завершается успешно, величина возвращаемого значения - не ноль.

Если функция завершается с ошибкой, величина возвращаемого значения - ноль. Чтобы получить дополнительные сведения об ошибке, вызовите GetLastError.

Если величина возвращаемого значения - не ноль, а число прочитанных байтов равняется нулю, указатель файла был за пределами текущего конца файла на момент операции чтения. Однако, если файл был открыт с флажком FILE\_FLAG\_OVERLAPPED, иlpOverlapped имеет  значение не ПУСТО (NULL), величина возвращаемого значения - ноль, а GetLastError возвращает ошибкуERROR\_HANDLE\_EOF, когда указатель файла проходит вне текущего конца файла.

Замечания

Если часть файла блокирована другим процессом, а операция чтения накладывается на блокированную часть, эта функция завершается ошибкой.

Когда работа с файлами открывается с флажком FILE\_FLAG\_NO\_BUFFERING, приложение должно соответствовать некоторым требованиям:

Доступ  к файлу должен начаться при смещениях байтов в пределах файла, которые должны быть целыми числами кратными размеру сектора тома. Чтобы выяснить размер сектора тома, вызовите функцию GetDiskFreeSpace.

Доступ  к файлу должен быть числом байтов, которое является целым числом кратным размеру сектора тома. Например, если размер сектора - 512 байтов, приложение может запросить для чтения и записи  512, 1024 или 2048 байтов, но не 335, 981 или 7171 байт.

Адреса буферов для операций чтения и записи должны выравниваться кратно сектору (выравниваются адреса в памяти, которые являются целыми числами, кратными размеру сектора тома). Один из способов выравнивания буфера кратно сектору, является использование функции VirtualAlloc, которая назначает буфера. Эта функция выделяет память, выравнивая их по адресам, которые являются целыми числами, кратными размеру страницы памяти  системы. Поскольку размеры и страницы памяти, и  сектора тома являются степенью 2, память, выровненная кратно размеру страницы памяти системы также выравнивается кратно и размеру сектора тома.

Доступ к буферу ввода, в то время как операция чтения использует этот буфер, может привести к искажению прочитанных данных в буфер. Приложения не должны читать из, записывать в, перераспределять или освобождать буфер ввода, который использует операция чтения  до тех пор, пока она не завершится.

Символы могут читаться из буфера ввода консоли при помощи использования функции ReadFile с дескриптором консольного ввода данных. Консольный режим работы обуславливает правильное поведение функции ReadFile.

Если именованный канал читается в режиме передачи сообщений, а длина следующего сообщения больше, чем это устанавливается параметром nNumberOfBytesToRead, ReadFile возвращает значение ЛОЖЬ (FALSE), а GetLastError возвращает ошибкуERROR\_MORE\_DATA. Остаток сообщения может быть прочитан последующим вызовом  функции ReadFile или PeekNamedPipe.

При чтении в устройстве обмена информацией (коммуникационном), поведение ReadFile управляется текущими коммуникационными задержками как установлено и возвращено обратно использованием функций [SetCommTimeouts](http://www.vsokovikov.narod.ru/New_MSDN_API/Comm_res/fn_setcommtimeouts.htm) и [GetCommTimeouts](http://www.vsokovikov.narod.ru/New_MSDN_API/Comm_res/fn_getcommtimeouts.htm). Непредсказуемые результаты могут произойти, если Вы будете не в состоянии установить значения времени простоя. Для получения дополнительной информации о коммуникационных блокировках по времени, см. статью о структуре [COMMTIMEOUTS](http://www.vsokovikov.narod.ru/New_MSDN_API/Comm_res/str_commtimeouts.htm).

Если ReadFile пытается читать из почтового слота, буфер которого является слишком маленьким, функция возвращает значение ЛОЖЬ (FALSE), а GetLastError возвращает ошибку ERROR\_INSUFFICIENT\_BUFFER.

Если дескриптор анонимного  канала записи был закрыт, а функция ReadFile пытается читать используя дескриптор соответствующего анонимного канала чтения, функция возвращает значение ЛОЖЬ (FALSE), а GetLastError возвращает ошибкуERROR\_BROKEN\_PIPE.

Функция ReadFile может завершиться ошибкой и возвратить значение ERROR\_INVALID\_USER\_BUFFER илиERROR\_NOT\_ENOUGH\_MEMORY всякий раз, когда имеется слишком много ожидающих обработки асинхронных запросов ввода - вывода.

Код проверки ReadFile положения метки конца файла (eof) различается для синхронных и асинхронных операций чтения.

Когда синхронная операция чтения достигает конца файла, ReadFile возвращает значение ИСТИНА (TRUE) и устанавливает в параметре \*lpNumberOfBytesRead  нуль. Нижеследующий типовой код проверяет метку конца файла для синхронной операции чтения:

|  |
| --- |
| // Делается попытка произвести синхронную операцию чтения.  bResult = ReadFile(hFile, &inBuffer, nBytesToRead,  &nBytesRead, NULL) ;  // Проверяем не конец ли это файла.  if (bResult && nBytesRead == 0, )  {  // Мы достигли конца файла.  } |

Асинхронная операция чтения может встретиться с концом файла в течение инициализации вызова  ReadFile, или в ходе последующей асинхронной операции.

Если EOF обнаруживается  функцией ReadFile во время асинхронной операции чтения, ReadFile возвращает значение ЛОЖЬ (FALSE), а GetLastError возвращает ошибку ERROR\_HANDLE\_EOF.

Если EOF обнаруживается в ходе последующей асинхронной операции, происходит вызов функции GetOverlappedResult, чтобы получить в результате этой операции возвращенное значение ЛОЖЬ (FALSE), а GetLastError возвратит ошибкуERROR\_HANDLE\_EOF.

Чтобы отменить  все ждущие обработки асинхронные операции ввода-вывода (I/O), используйте функцию [CancelIo](http://www.vsokovikov.narod.ru/New_MSDN_API/Menage_files/fn_cancelio.htm). Эта функция  отменяет только операции, порождаемые вызывающим потоком для заданного дескриптора файла. Отмененные операции ввода-вывода (I/O) завершаются ошибкой ERROR\_OPERATION\_ABORTED.

Если Вы пытаетесь читать из накопителя на гибких дисках, который не имеет дискеты, система показывает на экране окно сообщения, предлагающее пользователю повторить  операцию. Чтобы воспрепятствовать системе показывать это окно, вызовите функцию SetErrorMode с флажком SEM\_NOOPENFILEERRORBOX.

Пример кода

Нижеследующий типовой код иллюстрирует проверку на метку конца файла для асинхронной операции чтения:

|  |
| --- |
| OVERLAPPED gOverlapped;  // подготавливаем поля структуры асинхронной операции  gOverLapped.Offset = 0;  gOverLapped.OffsetHigh = 0;  gOverLapped.hEvent = hEvent;    // пытаемся провести асинхронную операцию чтения  bResult = ReadFile(hFile, &inBuffer, nBytesToRead, &nBytesRead,  &gOverlapped) ;    // если возникает проблема или асинхронная операция  // все еще ожидает обработки ...  if (!bResult)  {  // решаем что делать с кодом ошибки  switch (dwError = GetLastError())  {  case ERROR\_HANDLE\_EOF:  {  // мы достигли конца файла  // в течение вызова к ReadFile    // код обрабатывается так-то  }    case ERROR\_IO\_PENDING:  {  // асинхронный ввод-вывод все еще происходит    // сделаем кое-что пока он идет  GoDoSomethingElse() ;    // проверим результат работы асинхронного чтения  bResult = GetOverlappedResult(hFile, &gOverlapped,  &nBytesRead, FALSE) ;    // если возникла проблема ...  if (!bResult)  {  // решаем что делать с кодом ошибки  switch (dwError = GetLastError())  {  case ERROR\_HANDLE\_EOF:  {  // мы достигли конца файла  // в ходе асинхронной  // операции  }    // решаем что делать с другими случаями ошибок  }// конец процедуры switch (dwError = GetLastError())  }  } // конец процедуры case    // решаем что делать с другими случаями ошибок    } // конец процедуры switch (dwError = GetLastError())  } // конец процедуры if |