WM_QUERYENDSESSION и WM_ENDSESSION (Windows Shutdown)

Завершение работы Windows или как система завершает пользовательские приложения при Shutdown-е.

Здесь я хочу рассказать, что происходит, когда система завершает работу. Но рассматривается не весь процесс от начала до конца, а только та часть, когда система уведомляет работающие приложения о завершении работы (детальнее про остальные стадии можно почитать в [2]).

Функция Win32 API **ExitWindowsEx** передает запрос на завершение системы клиент/серверной подсистеме (csrss.exe) с помощью механизма RPC.

CSRSS (Client Server Runtime Process) перебирает все процессы в сеансе интерактивного пользователя в порядке убывания shutdown level.

Для всех процессов при создании shutdown level устанавливается в 640 ($280), но можно изменить с помощью функции **SetProcessShutdownParameters**. Shutdown level для Проводника равен 2, для Диспетчера задач – 1

Для каждого процесса, владеющего окном верхнего уровня, с помощью функции **SendMessageCallback** посылается сообщение WM_QUERYENDSESSION по очереди всем GUI-потокам.

Предположительно сообщение посылается в порядке создания потоков, т.е. если главный поток является GUI-шным, то он первым получит сообщение из всех потоков процесса).

Точнее – сообщение посылается окну верхнего уровня (top-level window), которым владеет поток. Если потоку принадлежит несколько таких окон, то сообщение посылается только одному из них. Таковым будет являться окно, не имеющего владельца, хэндл которого будет получен первым в коллбэке вызова **EnumThreadWindows**

Если поток возвращает True, завершение работы системы продолжается (т.е. путем возвращения False поток может остановить shutdown системы). Тогда CSRSS посылает потоку сообщение WM_ENDSESSION с требованием завершить свою работу.

Если поток успевает завершиться за отведенный ему таймаут, то CSRSS посылает пары сообщений WM_QUERYENDSESSION/WM_ENDSESSION следующему GUI-потоку процесса (если поток не владеет окном(ами) верхнего уровня, он пропускается). После того, как ссобщения будут обработаны всеми GUI-потоками

процесса и сам процесс завершится, CSRSS переходит к "обработке" следующего процесса в интерактивном сеансе.

Windows XP

Параметры, используемые CSRSS при завершении работы системы

[HKEY_CURRENT_USER\Control Panel\Desktop]

- **WaitToKillAppTimeout** (Milliseconds) - Влияет на время, в течении которого будет отображаться окно "Ожидание" (Рис.2.) с прогресс-баром. По-умолчанию 20000.
- **HungAppTimeout** (Milliseconds) - Определяет, как долго система ждет завершения потока, перед тем, как показать диалоговое окно "Зависание" (Рис.1.). По-умолчанию 5000.
- **AutoEndTasks** (BOOL) - Автоматически завершать зависшие приложения без показа диалогового окно "Зависание" (Рис.1.). По-умолчанию 0.

После отсылки потоку сообщения WM_QUERYENDSESSION/WM_ENDSESSION, CSRSS ждет завершения работы потока в течение времени, указанного в HungAppTimeout.

Ожидает CSRSS с помощью функции **MsgWaitForMultipleObjects** с обработкой всех сообщений на двух описателях: потока (окну которого было послано сообщение) и события отмены завершения работы системы

Если в течение указанного времени поток не завершается, CSRSS открывает одно из диалоговых окон, которые мы условно назовем "Зависание" (Hung) и "Ожидание" (Wait). На самом деле диалог один и тот же, просто в зависимости от некоторых обстоятельств меняется ого содержимое). Похоже вид диалога зависит от состояния потока, окну которого было отправлено сообщение - завис он или находится в состоянии ожидания. Все это справедливо для систем до Висты.

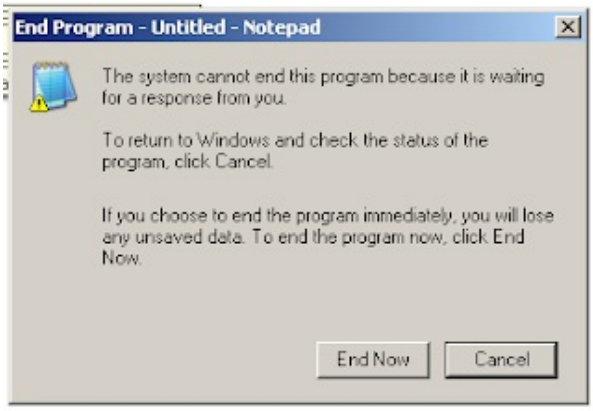


Рис.1. "Зависание" (Windows XP)

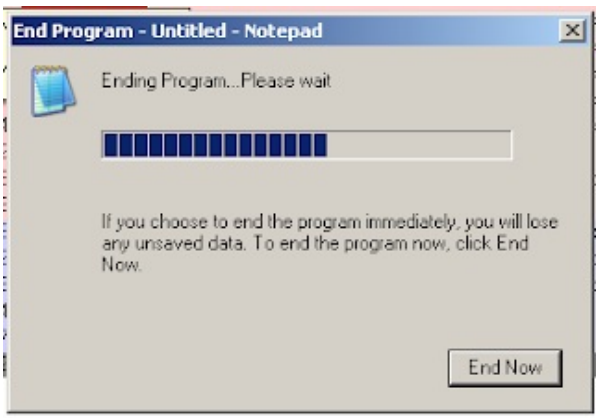


Рис.2. "Ожидание" (Windows XP)

Причем диалоговое окно "Ожидание", когда прогресс-бар добегает до конца, "превращается" в окно "Зависание". Время, за которое прогресс-бар побежит до конца зависит от значения параметров WaitToKillAppTimeout и HungAppTimeout.

Чтобы в UI все это выглядело корректно, должно выполняться условие:
WaitToKillAppTimeout > HungAppTimeout

Тем самым пользователь уведомляется, что корректное завершение данной программы невозможно, и предлагает принудительно завершить процесс либо отметить завершение работы системы (таймаута для этого окна не предусмотрено, т.е. на этом этапе запрос на завершение процесса может ждать бесконечно долго).

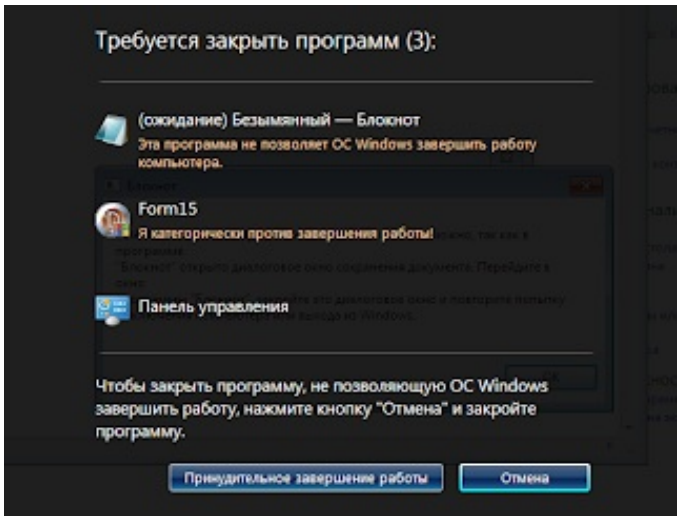
Если же AutoEndTasks = 1, диалоговое окно "Зависание" не отображается – вместо этого процесс автоматически завершается (TerminateProcess).

Windows Vista/7

Окно информирования пользователя о программах, мешающих выключению, претерпело изменения – теперь визуального отличия между "зависанием" и "ожиданием" нет.

Вместо системного диалога в Windows Vista/7 отображается полноэкранный UI, который более четко передает информацию о приложениях, блокирующих выключение. Кроме того, система способна определить несколько приложений, блокирующих завершение работы (с помощью новых API) и причины этой блокировки.

Рис.3. Программа не позволяет завершить сеанс (Windows 7)



В Windows XP диалог для блокирующего приложения позволяет пользователям отменить выключение или принудительно завершить "зависшее" приложение. И такой диалого отображается для каждого "зависшего" приложения. Это не очень удобно для многих пользователей, которые "просто хотят, чтобы из компьютер выключился".

При выключении компьютера, Windows VistaVista/7 решает эту проблему, разрешая пользователям завершить все приложения, блокирующие выключение ("силовой" shutdown). В таком варианте, Windows будет отправлять приложениям сообщение WM_QUERYENDSESSION с флагом ENDSESSION_FORCEFULSHUTDOWN. Если приложение ответит FALSE, система продолжит завершение работы, вместо того, чтобы отметить его и отправит сообщение WM_ENDSESSION. Если же приложение не ответить на сообщение WM_QUERYENDSESSION/WM_ENDSESSION за отведенный ему таймаут, Windows принудительно завершить его.

В Windows XP, приложения имеют право отклонить WM_QUERYENDSESSION без какого-либо отображения UI и указания причины, почему они отменили выключение. Windows Vista/7 в любом случае отображает UI, даже если в ответ WM_QUERYENDSESSION было получено FALSE.

В Windows Vista/7 появился новый механизм, который позволяет приложениям "на лету" регистрировать строковое сообщение, поясняющее причину, по которой они блокируют завершение работы системы. И, соответственно, отменять регистрацию, если в блокировании shutdown-а больше нет необходимости.

При выключении компьютера, Windows Vista/7 определяет зависшие приложения и автоматически завершает их.

Зависшим считается приложение, которое не ответило ни на одно свое оконное сообщенные за последние 5 секунд.

Варианты поведения системы

#	**Описание**
[1]	Поведение, аналогичное Windows XP
[2]	Изменено по сравнению с Windows XP. Приложение не завершается
[3]	Изменено по сравнению с Windows XP. Система может принудительно завершить приложение

Нормальное завершение работы

Сообщение	Есть видимое окно верхнего уровня или задана "reason string"	Нет видимого окна верхнего уровня и не задана "reason string"
WM_QUERYENDSESSION	У приложения есть столько времени, сколько ему необходимо, чтобы ответить на WM_QUERYENDSESSION. Через 5 секунд система покажет UI (Рис.3) [1]	У приложения есть 5 секунд на то, чтобы ответить на WM_QUERYENDSESSION, затем Windows завершит его, если приложение не отвечает [3]
WM_QUERYENDSESSION	Windows показывает UI (Рис.3), если приложение вернуло FALSE в ответ на WM_QUERYENDSESSION [2]	Система посылает приложению сообщение WM_ENDSESSION, если в ответ на WM_QUERYENDSESSION оно вернуло FALSE [3]
WM_ENDSESSION	У приложения есть столько времени, сколько ему необходимо, чтобы ответить на WM_ENDSESSION. Через 5 секунд система покажет UI (Рис.3) [1]	У приложения есть 5 секунд на то, чтобы ответить на WM_ENDSESSION, затем Windows завершит его, если приложение не отвечает [3]

Силовое завершение работы

Сообщение	Есть видимое окно верхнего уровня или задана "reason string"	Нет видимого окна верхнего уровня и не задана "reason string"

WM_QUERYENDSESSION	У приложения есть 1 секунда на то, чтобы ответить на WM_QUERYENDSESSION, затем Windows завершит его, если приложение не отвечает [3]	У приложения есть 1 секунда на то, чтобы ответить на WM_QUERYENDSESSION, затем Windows завершит его, если приложение не отвечает [3]
WM_QUERYENDSESSION	Система посылает приложению сообщение WM_ENDSESSION, если в ответ на WM_QUERYENDSESSION оно вернуло FALSE [2]	Система посылает приложению сообщение WM_ENDSESSION, если в ответ на WM_QUERYENDSESSION оно вернуло FALSE [2]
WM_ENDSESSION	У приложения есть 30 секунд на то, чтобы ответить на WM_ENDSESSION, затем Windows завершит его, если приложение не отвечает [3]	У приложения есть 30 секунд на то, чтобы ответить на WM_ENDSESSION, затем Windows завершит его, если приложение не отвечает [3]

Использование Shutdown Reason API (новое в Windows Vista/7)

Объявления функций (Delphi)

```
function ShutdownBlockReasonCreate(Wnd: HWND; const pwszReason: PWideChar): BOOL;
stdcall;
function ShutdownBlockReasonDestroy(Wnd: HWND): BOOL; stdcall;
function ShutdownBlockReasonQuery(Wnd: HWND; pwszBuff: PWideChar;
  var pcchBuff: DWORD): BOOL; stdcall;
```

Microsoft рекомендует в случае, если приложение должно заблокировать выключение компьютера, использовать это API

1. Если приложению надо заблокировать завершение работы, оно должно вызвать ShutdownBlockReasonCreate для регистрации строки-причины, передав хэндл окна, которое будет обрабатывать WM_QUERYENDSESSION
2. Когда у приложения больше не будет причин для блокирования shutdown-а, оно должно вызвать ShutdownBlockReasonDestroy для отмены регистрации строки-причины.
3. ShutdownBlockReasonQuery используется для получения строки-причины, зарегистрированной ранее

Ссылки

[1] Application Shutdown Changes in Windows Vista
[2] How Windows Shuts Down