Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Дисциплина: Системное программное обеспечение ВМ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту

на тему

Плагин для Total Commander «Диспетчер задач»

БГУИР КП 1-40 02 01 19\* ПЗ

Выполнил:

Студент гр. 550501 Попов М.С.

Проверил: Лавникевич Д. А.

Минск 2017

**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ………………………………………………………………………..4

1 ПОСТАНОВКА ЦЕЛИ И ЗАДАЧИ……………..……………………………..6

2 ОБЗОР ИСТОЧНИКОВ…………...……………..……………………………..7

2.1 Обзор аналогов……………………………………………………………..7

2.2 Инструменты для реализации приложения……………..………………..7

3 РАЗРАБОТКА АЛГОРИТМОВ…..……………..……………………………..8

3.1 Схемы алгоритмов…………………………………………………………8

4 ОПИСАНИЕ ФУНКЦИЙ……..…..……………..……………………………..9

5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ…………..……………………………..16

6 РЕЗУЛЬТАТЫ РАБОТЫ……..…..……………..………………………….....17

7 КОД ПРОГРАММЫ…………..…..……………..…………………………….20

ЗАКЛЮЧЕНИЕ…… …………...…..……………..………………..…………..21

ЛИТЕРАТУРА...…………..……..…..……………..…………………...……….23

**ВВЕДЕНИЕ**

Файловый менеджер Total Commander – это, пожалуй, одна из самых удачных программ, которая была написана для Windows. Практически не изменяя свой внешний вид, она эволюционирует вместе с операционной системой Windows, оставаясь для многих системной утилитой номер один. Во многом, это большая заслуга тех программистов, которые смогли увидеть, насколько перспективным может оказаться принцип двухпанельного интерфейса файлового менеджера. За более чем двадцать пять лет разработки, продукт компании Ghisler оброс множеством самых разнообразных функций: появились закладки, добавилась поддержка rar-архивов, менеджер стал работать с юникодом и стал во многом удобнее. Но, как известно, совершенству нет предела, и, памятуя эту истину, разработчики переложили «доработку» программы на плечи самих пользователей. С помощью дополнительных модулей или попросту говоря плагинов, Total Commander можно превратить в многофункциональный швейцарский нож, в котором есть инструменты на все случаи жизни. Причем, некоторые плагины не только расширяют потенциал программы, но, даже могут послужить поводом для того, чтобы отказаться от некоторых программ, которыми ранее приходилось пользоваться.

Все дополнительные модули для Total Commander делятся на несколько групп:

* Плагины для архивации – дополнительные модули, которые расширяют в программе возможности по работе с архивами. Такие плагины имеют расширение WCX;
* Плагины файловой системы – эти дополнения к Total Commander позволяют получить доступ к дискам или другим цифровым носителям с файловой системой, которая не поддерживается Windows. Эти плагины выглядят как файлы с расширением WFX;
* Плагины внутреннего просмотрщика – предназначены для расширения списка форматов, поддерживаемых внутренним просмотрщиком Lister. Эти файлы имеют расширение WLX;
* Информационные плагины – дают возможность просмотреть информацию, заложенную в файл. Они имеют расширение WDX.

Для решения поставленной в данном курсовом проекте задачи, будет создан плагин файловой системы, т.к.  данный тип плагина обеспечивает доступ к части компьютера, куда нет доступа через имена дисков, а именно, к запущенным на данный момент процессам. А процессы, это то, что в первую очередь должно быть исследовано плагином «Диспетчер задач».

**1 ПОСТАНОВКА ЦЕЛИ И ЗАДАЧИ**

Цели курсовой работы: освоение практических навыков разработки и проектирования полноценного, отлаженного программного продукта; закрепление и углубление теоретических знаний, полученных при изучении курса «Системное программное обеспечение ВМ»; изучить и использовать сторонние технологии и средства.

Задача курсовой работы: проектирование плагина «Диспетчер задач» для Total Commander. Данный плагин должен выводить на экран список запущенных процессов  и основную информацию о каждом из них (в частности, идентификатор процесса,  время его запуска  и текущее использование памяти). Также предусмотрена возможность отображения почти полной информации о работающем процессе, а еще возможность снятия с выполнения выбранного процесса (вплоть до принудительного системного снятия).

**2 ОБЗОР ИСТОЧНИКОВ**

**2.1 Обзор аналогов**

Основным аналогом, который помог при написании данного плагина, является Windows Task Manager(диспетчер задач). Это один из самых удобных и полезных инструментов в Windows, предназначенный для диагностики и управления процессами в системе. В диспетчере задач отображаются сведения о программах и процессах, выполняемых на компьютере. Кроме того, в нем можно просмотреть наиболее часто используемые показатели быстродействия процессов.

Хорошими аналогами также являются плагины AceHelper и ProcFS.

Но информация о процессах, которую они предоставляют пользователям показалась мне не достаточно полной, что заставило меня задуматься о решении этого вопроса в своем плагине.

**2.2 Инструменты для реализации приложения**

При выборе средств для разработки плагина необходимо было учесть множество различных аспектов, наиболее важным из которых является язык программирования. Именно он определяет возможности реализации тех или иных программных решений.

Основными языками для написания плагинов для Total Commander являются С#,Visual Basic, C++ и Delphi. Долго выбирать не пришлось: язык С++ я нашел более мощным инструментом для решения поставленной задачи. Тем более я имел достаточно опыта работы с этим универсальным языком.

Для разработки данного плагина используется среда Visual Studio 2013.

WinAPI и Total Сommander plugin API также задействованы для реализации всего задуманного функционала.

**3 РАЗРАБОТКА АЛГОРИТМОВ**

**3.1 Схемы алгоритмов**

3.1.1 Алгоритм функции **DWORD GetProcessByExeName(char \*ExeName)**,где ExeName – это имя процесса. Данная функция ищет процесс по имени и возвращает его идентификатор. Схема алгоритма представлена в Приложении A (Схема алгоритма **DWORD GetProcessByExeName(char \*ExeName)**).

3.1.2 Алгоритм функции **BOOL FsDeleteFileW(WCHAR\* RemoteName)**, где RemoteName – имя процесса, который необходимо удалить. Данная функция удаляет процесс по имени. Схема алгоритма представлена в Приложении Б (Схема алгоритма **BOOL FsDeleteFileW(WCHAR\* RemoteName)**).

3.1.3 Алгоритм функции **WCHAR\* Slesh(char\* str,WCHAR\* res)**,где str – путь к процессу, res – обработанный путь с двойными обратными слэшами. Данная функция добавляет по одному дополнительному слешу к каждому слешу в пути к exe–файлу, который запустил процесс. Схема алгоритма представлена в Приложении В (Схема алгоритма **WCHAR\* Slesh(char\* str,WCHAR\* res)**).

**4 ОПИСАНИЕ ФУНКЦИЙ**

Для функционирования любого системного плагина Total Commander обязательно необходимы следующие функции: FsInitW, FsFindFirstW, FsFindNextW, FsFindCloseW.

1) Int FsInitW(int PlNr, tProgressProc tP,tLogProc tL, tRquestProc tR);

Данная функция вызывается при загрузке плагина. Все переданные в нее значения сохраняются для дальнейшего использования в программе.

Описание параметров:

* PlNr - внутренний номер, который выдал Total Commander данному плагину;
* tP - указатель на функцию обратного вызова;
* tL – указатель на функцию авторизации;
* tR – указатель на процедуру запроса текста;

Возвращаемое значение не используется.

2) HANDLE FsFindFirstW(char\* Path,WIN32\_FIND\_DATA \*FindData);

Данная функция вызывается для извлечения первого процесса в каталог плагина.

Описание параметров:

* Path - полный путь к exe файлу, который запустил процесс;
* FindData – структура для хранения сведений о файле;

Возвращает снимок всех процессов, который будет передаваться как параметр в функцию FsFindNextW.

3)BOOL FsFindNextW(HANDLE Hdl,WIN32\_FIND\_DATA \*FD);

Данная функция, вызывается после того как отработает FsFindFirstW, которая вернет снимок процессов. Далее, используя этот снимок, функция извлечет оставшиеся процессы в каталог плагина.

Описание параметров:

* Hdl – снимок всех процессов;
* FD - структура для хранения сведений о файле;

Возвращает TRUE, если найден новый процесс, и FALSE если больше процессов нет. После возвращения FALSE , поиск процессов завершится и сработает функция FsFindCloseW.

4) int FsFindCloseW(HANDLE Hdl);

Вызывается для того, чтобы завершить цикл работы функции FsFindNextW.

Описание параметров:

* Hdl – дескриптор последнего найденного процесса;

Возвращаемое значение не используется.

5) int FsExecuteFileW(HWND MW,WCHAR\* RemoteName,WCHAR\* Verb);

Вызывается, чтобы выполнить файл в файловой системе плагина или показать его список свойств. В случае данного плагина в списке свойств определены только два действия: удалить и показать свойства.

Описание параметров:

* MW - родительское окно, которое можно использовать для отображения листа свойств;
* RemoteName - имя исполняемого файла с полным путем;
* Verb – действие из списка операций, например: “удалить”, “свойства” и т.д;

Возвращает FS\_EXEC\_OK, если команда была успешно выполнена в плагине, и FS\_EXEC\_ERROR, если команда не выполнена.

6) BOOL FsDeleteFileW(char\* RemoteName);

Вызывается для того, чтобы завершить любой процесс.

Описание параметров:

* RemoteName -имя процесса, который нужно удалить, с полным путем;

Возвращает TRUE, если файл можно удалить, FALSE, если нет.

7) void FsGetDefRootName(char\* DefRootName,int maxlen);

Вызывается только при установке плагина. Функция определяет имя плагина, которое будет отображаться в Network Neighborhood разделе Total Commander.

Описание параметров:

* DefRootName – имя плагина;
* maxlen – максимальное количесвто символов;

Функция ничего не возвращает.

8)int FsExtractCustomIconW(char\* RemoteName,int ExtFlags,HICON\* TheIcon);

Вызывается для отображения пользовательских значков процессов при выводе списка на экран.

Описание параметров:

* RemoteName – это полный путь к процессу, значок которого должен быть извлечен;
* ExtFlags - флаги для операции извлечения;
* TheIcon - дескриптор значка;

Функция возвращает либо FS\_ICON\_EXTRACTED\_DESTROY ,либо

FS\_ICON\_USEDEFAULT , в зависимости от того найдена ли у процесса иконка или нет.

9) int FsContentGetSupportedField(int FIndex,char\* FName, char\* Un, int max);

Вызывается для перечисления всех поддерживаемых полей пользовательских колонок Total Commander.

Описание параметров:

* FIndex - индекс поля, для которого TC запрашивает информацию;
* FName - имя поля с индексом FIndex;
* Un – единицы измерения(в случае данного плагина они не используются);
* max – максимальное количество символов, которое может быть возвращено в каждом из полей.

Функция возвращает тип поля(t\_nomorefields, ft\_numeric\_32 , ft\_numeric\_64 , ft\_date , ft\_time и т.д.).

10) int \_\_stdcall FsContentGetValueW(char\* FileName,int FieldIndex,int UnitIndex, void\* FieldValue,int maxlen,int flags);

Вызывается для извлечения значения определенного поля для заданного процесса.

Описание параметров:

* FileName - Имя процесса, для которого плагин должен вернуть данные поля;
* FieldIndex - Индекс поля, для которого должен быть возвращен контент;
* UnitIndex - индекс используемой единицы измерения.
* FieldValue - здесь плагин должен вернуть запрошенные данные. Формат данных зависит от типа поля;
* maxlen - максимальное количество байтов, вписывающихся в переменную FieldValue.

Функция возвращает тип поля(t\_nomorefields, ft\_numeric\_32 , ft\_numeric\_64 , ft\_date , ft\_time и т.д.) или ft\_fileerror, что значит, что возникла ошибка доступа к процессу.

11) int FsContentGetDefaultSortOrder(int FieldIndex);

Вызывается, когда пользователь нажимает на заголовок сортировки над столбцами.

Описание параметров:

* FieldIndex - индекс поля, для которого должен быть возвращен порядок сортировки;

Возвращает 1 для возрастания (a..z, 1..9) или -1 для убывания (z..a, 9..0).

12)BOOL FsContentGetDefaultView(char\* ViewContents,char\* VHeaders,char\* ViewWidths,char\* ViewOptions,int maxlen);

Вызывается, чтобы получить внешний вид по умолчанию плагина(пользовательских колонок, их названия и ширину), к которым Total Commander должен переключаться при открытии плагина.

Описание параметров:

* ViewContents – поле по умолчанию для этого плагина;
* ViewContents - заголовок колонки;
* ViewWidths – ширина колонки;

Возвращает TRUE, если удалось вернуть представление колонок по умолчанию, FALSE, если ничего не получилось.

13) WCHAR\* Slesh(CHAR\* str, WCHAR\* res)

Вызывается для того, чтобы добавить дополнительные обратные слэшы

ко всем обратным слэшам в пути к exe-файлу, который запустил процесс.

Описание параметров:

* str – путь к exe-файлу, запустевшему процесс;
* res – обработанный путь с двойными обратными слэшами;

Возвращает обработанный путь с двойными обратными слэшами.

14) BOOL EnableDebugPrivilege(BOOL bEnable);

Вызывается для того, чтобы открыть доступ к работе с некоторыми защищенными процессами.

Описание параметров:

* bEnable – true разрешает доступ, false запрещает;

Возвращает TRUE, если удалось разрешить доступ к системным процессам, FALSE – если нет.

15) DWORD GetProcessByExeName(char \*ExeName) ;

Вызывается для того, чтобы получить идентификатор процесса по его имени.

Описание параметров:

* ExeName – имя искомого процесса;

Возвращает идентификатор процесса, если он найден. В случае, когда искомого процесса не существует, функция возвращает NULL;

16) int showSV(DWORD pid, char\* name);

Вызывается для того, чтобы вывести форму с дополнительными свойствами процесса.

Описание параметров:

* pid – идентификатор процесса;
* name – имя процесса;

Возвращает код выхода из приложения.

**5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

Для использования данного плагина необходим Total Commander версии 8.50 или выше (с поддержкой 64-битной версии плагина) и операционная система: 64-разрядные версии Windows.

Для того, чтобы установить данный плагин требуется:

1)Открыть Total Commander.

2)Теперь необходимо открыть настройки Total Commander. (Конфигурация - Настройка - Плагины / Configuration - Options - Plugins).

3)Ищем кнопку “Настройка” в пункте “Плагины файловой системы(.WFX)” и нажимаем на нее.

4)Теперь нажимаем “Добавить”(“Add”) и выбираем файл с расширением .wfx - собственно файл устанавливаемого плагина(fsplugin64.wfx). Все! Плагин установлен, найти его теперь можно в разделе “Сетевое окружение” Total Commander.

**6 РЕЗУЛЬТАТЫ РАБОТЫ**

В данном разделе приведем небольшой обзор работы плагина “Диспетчер задач” для Total Commander.

После установки плагина (установка описана в предыдущем разделе) он должен появиться в разделе Total Commander “Сетевое окружение”. Используя ресурсный файл, плагину была передана соответствующая иконка, а за передачу имени отвечает специальная функция, описанная в пункте 4(рис.6.1.).

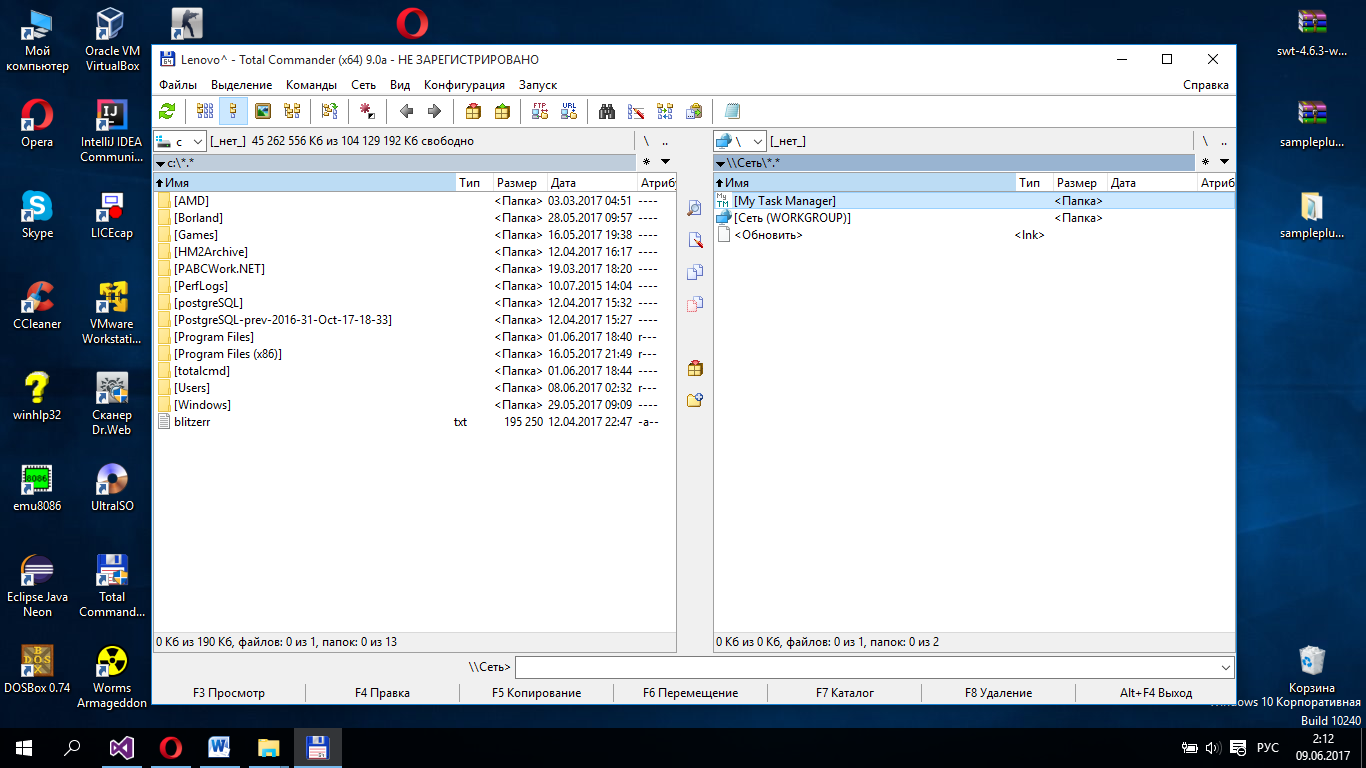


Рис.6.1. Раздел “Сетевое окружение”

После открытия плагина перед пользователем отображается список всех запущенных процессов. В первой колонке находится имя процесса. Вторая колонка, является стандартной колонкой Total Commander для плагинов файловой системы и никакой полезной информации в данном случае не несет. Третья колонка содержит идентификационный номер запущенного процесса. Четвертая и пятая колонки хранят значение текущей используемой памяти (в килобайтах) и время старта процесса соответственно (рис.6.2.).

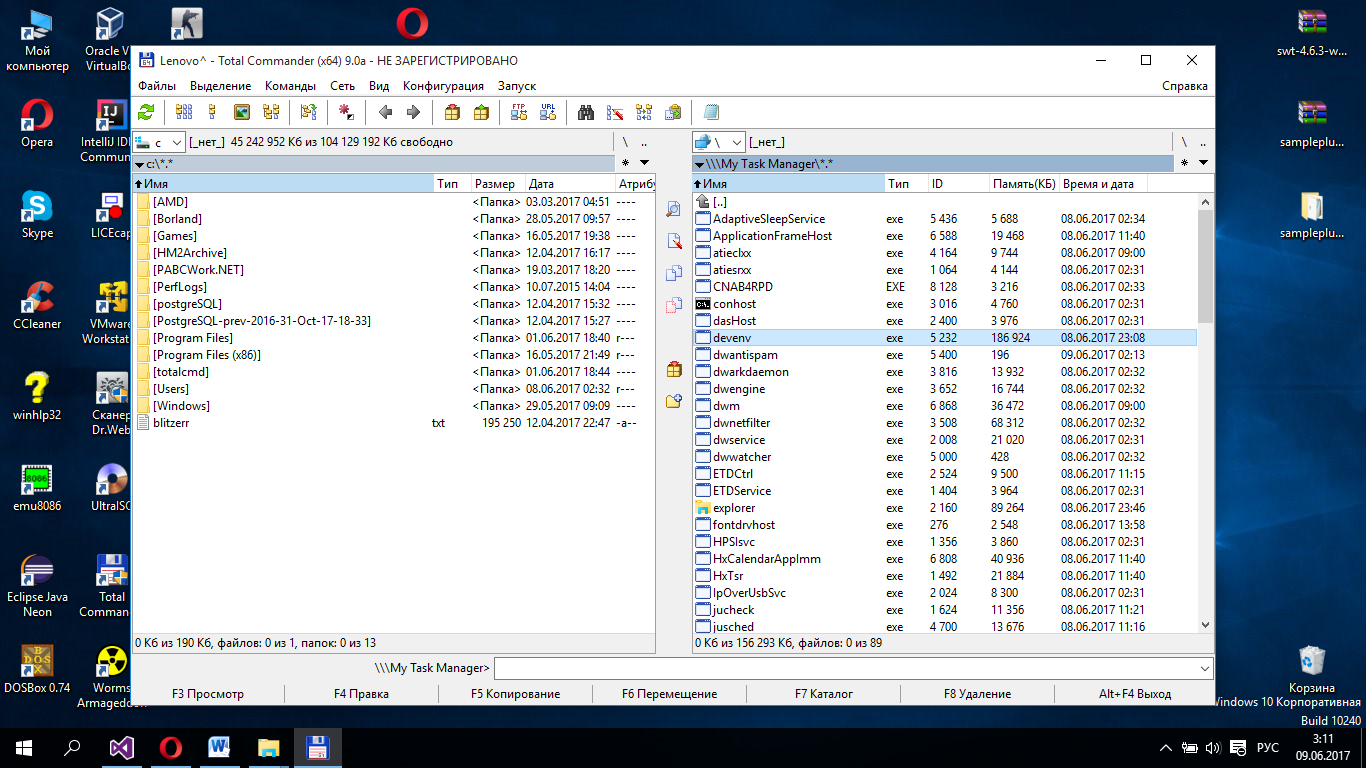


Рис. 6.2. Запущенный плагин “Диспетчер задач”

Для того, чтобы получить более детальные характеристики процесса, необходимо нажать на интересующий нас процесс правой кнопкой мыши и выбрать в открывшемся списке “Свойства” (рис.6.3.).

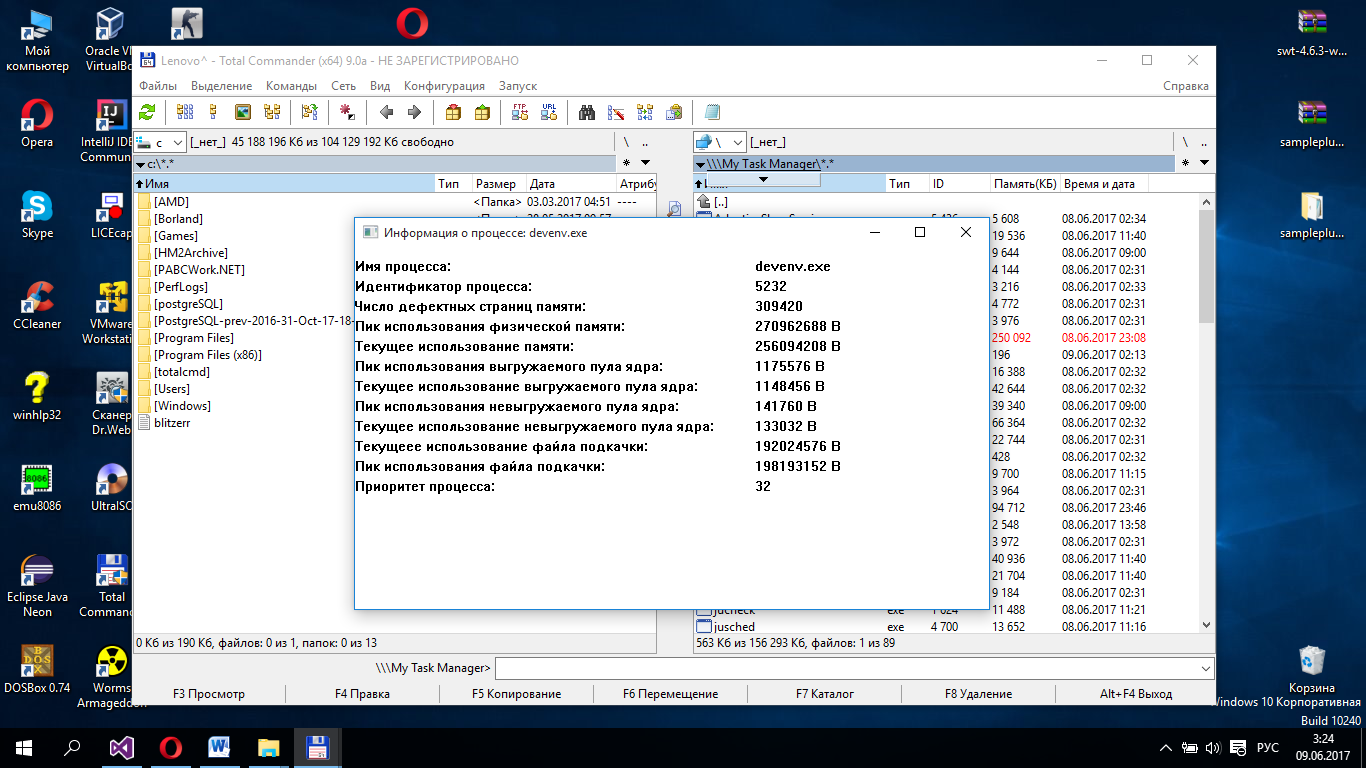


Рис. 6.3. Детальные свойства процесса

Также данный плагин способен завершить любой процесс. Для того, чтобы это сделать, нужно нажать на процесс правой кнопкой мыши и выбрать “Удалить”. После этого специальное сообщение оповестит пользователя об успешном выполнении операции или об обратном (рис.6.4.).

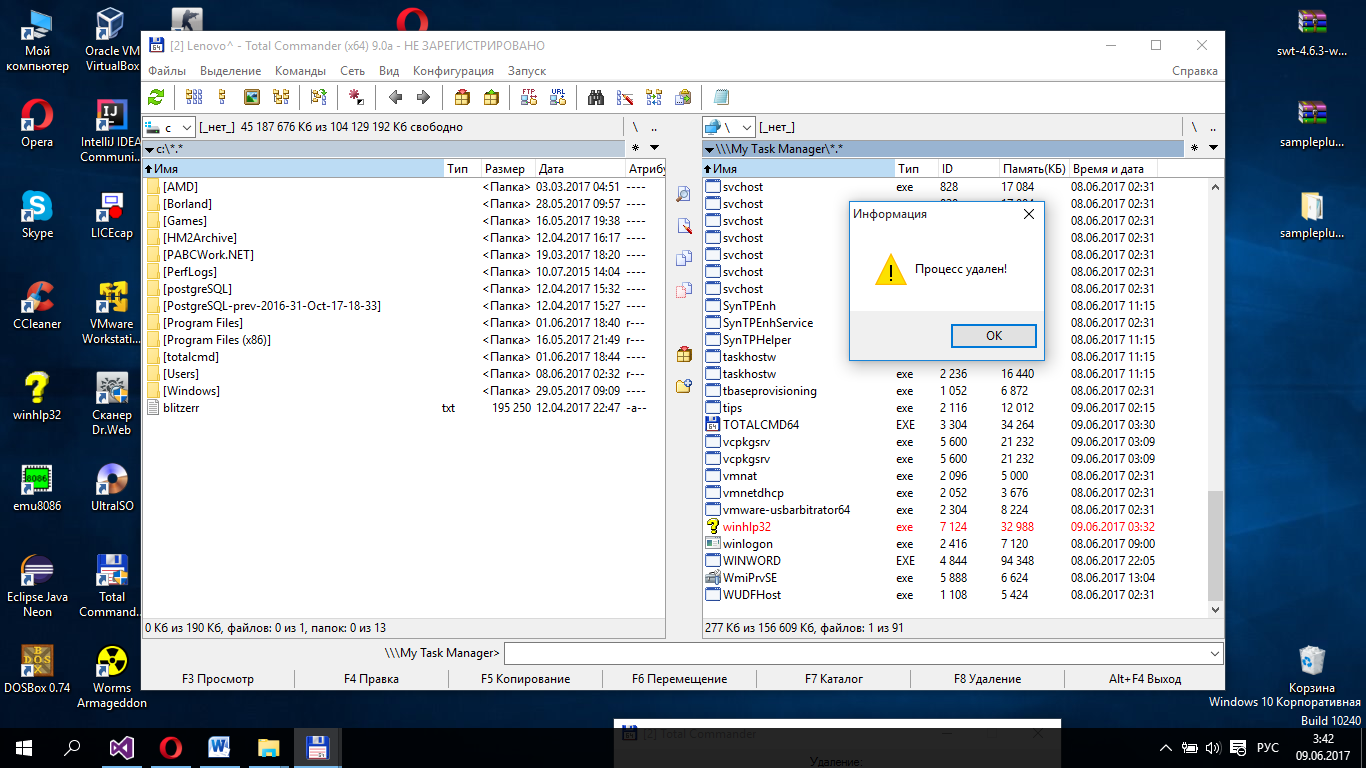


Рис. 6.4. Удаление процесса

**7 КОД ПРОГРАММЫ**

Код программы можно скачать по ссылке: https://github.com/WhyNOTnow/Course-work

**ЗАКЛЮЧЕНИЕ**

В данном курсовом проекте был разработан плагин “Диспетчер задач” для Total Commander. В процессе выполнения данного курсового проекта был получен опыт работы с WinAPI и Total Commander Plugin API. Так как основными объектами исследований стали запущенные процессы, знания об этом моменте программирования перешли на более высокий уровень.

Хоть Total Commander и теряет сейчас свою популярность, данная программа до сих пор остается достаточно мощной системной утилитой, способной во много раз расширить свой функционал, используя различные плагины.

Полученное приложение обладает простым интерфейсом. Пользовательские колонки содержат всю необходимую информацию о запущенных процессах. При необходимости можно открыть более подробную информацию о процессе, что отличает данный плагин от некоторых его аналогов. Разумеется, предусмотрена и одна их самых важных функция “Диспетчера задач”, а именно удаление процессво.

Плагин может быть использован в различных целях:

1)Мониторинг загруженности системы;

2)Отслеживание “жадных” процессов, которые слишком сильно загружают систему;

3)Удаление процессов, которые не получается закрыть обычным образом ввиду каких-либо проблем работы системы;

И многое другое.

В дальнейшем данный плагин будет улучшаться. Планируется расширить функционал и добавить некоторые дополнительные возможности, такие как история воздействия процесса на систему, контроль “жадных” процессов, возможность задавать приоритет процессам.

**ЛИТЕРАТУРА**

1. Все о Total Commander https://wincmd.ru

# 2. Введение в Win32 API http://cppstudio.com/post/9384/

3. Writing file system plugins (version 2.1 SE) for Total Commander

Copyright (C) 2002-2011 Christian Ghisler

4.Получаем список запущенных процессов http://eax.me/winapi-process-list/