**Лабораторная работа №4**

**ЧАСТЬ 1. УПРАВЛЕНИЕ ПРОЦЕССАМИ И СЛУЖБАМИ   
ИЗ КОМАНДНОЙ ОБОЛОЧКИ MICROSOFT POWERSHELL**

Для обеспечения стабильной и эффективной работы операционной системы большое значение имеет мониторинг работы приложений (запущенных процессов) и настройка функционирования служб (процессов, запускаемых в фоновом режиме). Рассмотрим вопросы, связанные с управлением процессами и службами из оболочки PowerShell.

**УПРАВЛЕНИЕ ПРОЦЕССАМИ**

Основным графическим инструментом для управления процессами на локальной машине является Диспетчер задач Windows (для запуска Диспетчера задач можно нажать комбинацию клавиш <Ctrl>+<Shift>+<Esc>), который отображает информацию о выполняющихся процессах, позволяет останавливать их и задавать приоритет выполнения (рис. 1)

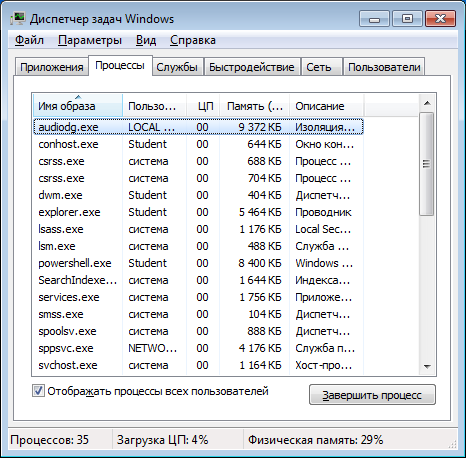


Рис. 1.

Начиная с версии Windows XP, в поставки операционных систем включены утилиты командной строки tasklist (просмотр списка процессов, запущенных на локальном или удаленном компьютере) и taskkill (остановка процесса). Этими утилитами можно пользоваться и в PowerShell (рис. 2), но эта среда предоставляет и более удобные встроенные средства. Рассмотрим способы решения с помощью PowerShell некоторых типичных задач, связанных с процессами.

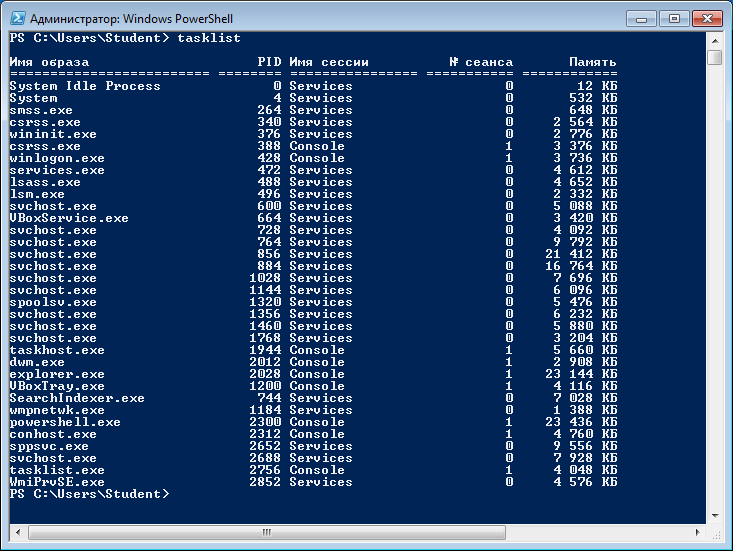


Рис. 2.

**Просмотр списка процессов**

В PowerShell получить список запущенных процессов позволяет командлет   
Get-Process. Если запустить этот командлет без параметров, то на экран будет выведена информация обо всех запущенных процессах (рис. 3):

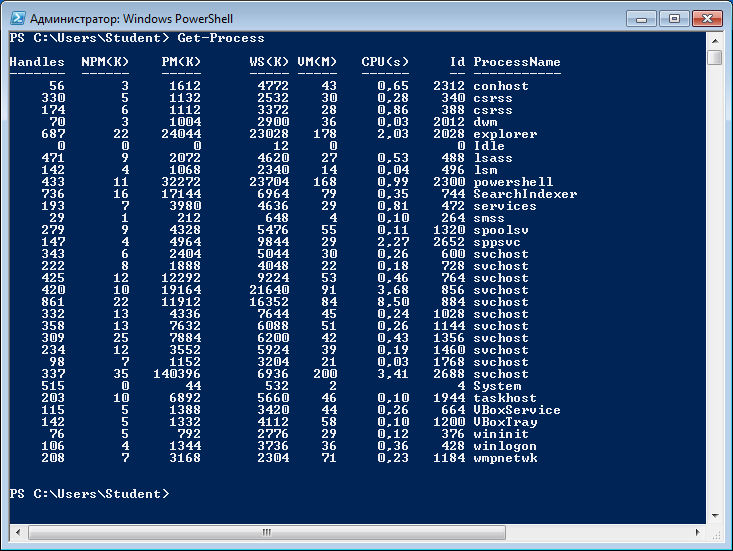


Рис. 3

Каждому процессу соответствует объект типа System.Diagnostics.Process; командлет Get-Process по умолчанию отображает несколько ***свойств этих объектов***, в частности:

* Handles – Счетчик дескрипторов (свойство HandleCount объекта System.Diagnostics.Process);
* PM – Объем используемой виртуальной памяти (свойство PagedSystemMemorySize объекта System.Diagnostics.Process);
* WS – Объем используемой памяти (свойство WorkingSet объекта System.Diagnostics.Process);
* CPU – Количество процессорного времени, затраченное процессом (свойство TotalProcessorTime.TotalSeconds объекта System.Diagnostics.Process);
* ID – Идентификатор процесса (свойство Id объекта System.Diagnostics.Process);
* ProcessName – Имя процесса (свойство ProcessName объекта System.Diagnostics.Process).

Можно вывести информацию об одном или нескольких процессах с определенными именами (при этом в именах можно применять подстановочные символы). Например, следующая команда выведет все процессы, имена которых начинаются на букву "s" (рис. 4):

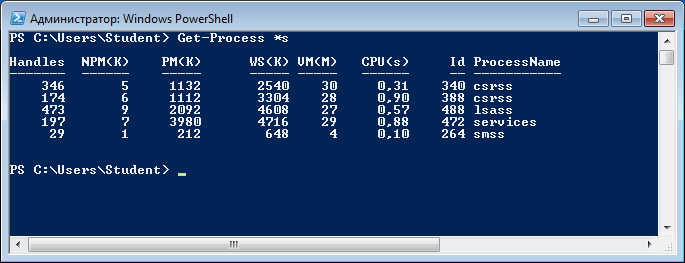


Рис. 4

Для сортировки или фильтрации списка процессов применяются, как обычно, командлеты Sort-Object, Where-Object и Select-Object.

Например, конвейер команд Get-Process | Sort-Object WS -Descending выведет на экран все процессы, отсортированные по объему используемой памяти (рис. 5), а конвейер Get-Process | Select-Object -First 5 – только 5 процессов из общего списка процессов (рис. 6).

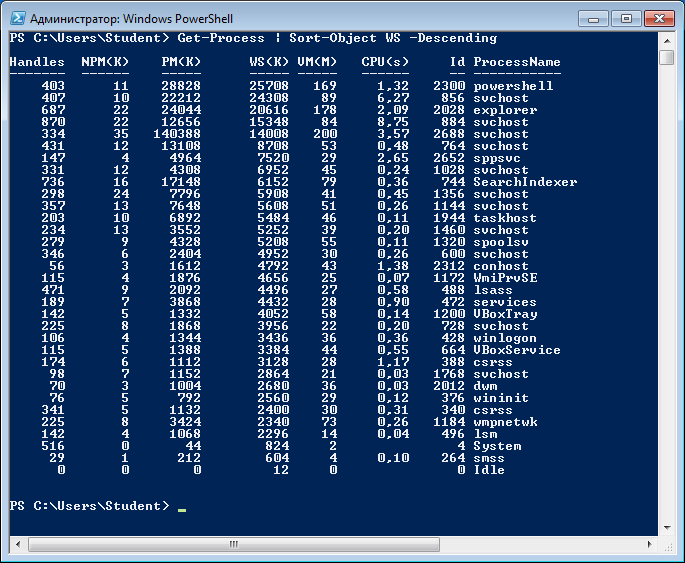


Рис. 5

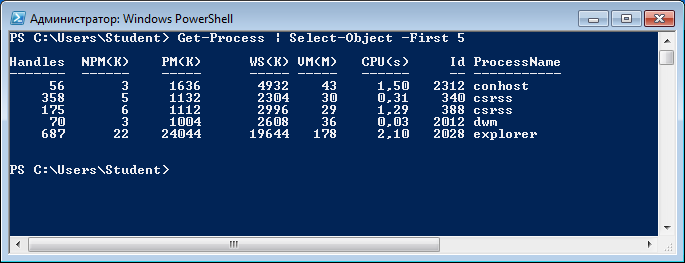


Рис. 6.

**Остановка процессов**

Остановить процесс на локальной машине позволяет командлет Stop-Process, имеющий псевдоним kill. При этом по умолчанию используется параметр Id, требующий указания идентификатора останавливаемого процесса (напомним, что идентификатор процесса можно узнать с помощью командлета Get-Process).

Например, следующая команда остановит все процессы с именем notepad:   
PS C:\Users\Student> Stop-Process -Name notepad (рис. 7)

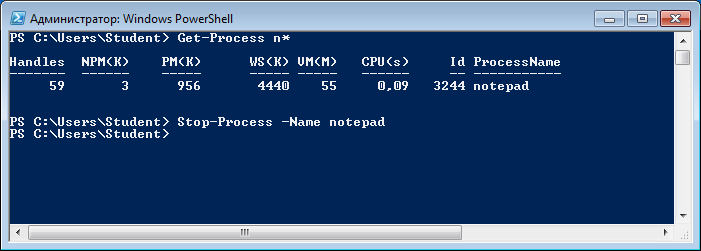


Рис. 7.

Для остановки процессов с выводом информации нужно указать параметр   
-PassThru, например,

PS C:\Users\Student> Stop-Process -Name notepad –PassThru (рис. 8):

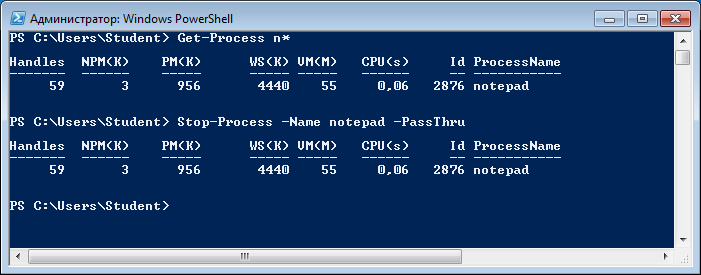


Рис. 8.

**Запуск процессов**

Для запуска процесса из оболочки PowerShell можно просто указать путь к соответствующему исполняемому файлу, например:   
PS C:\Users\Student> C:\WINDOWS\system32\calc.exe (рис. 9).

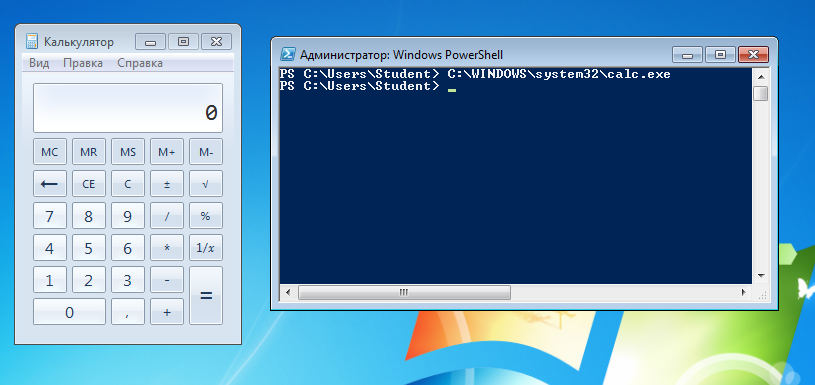


Рис. 9.

Параметр -Confirm командлета Stop-Process включает режим подтверждения при остановке процессов (рис. 10).

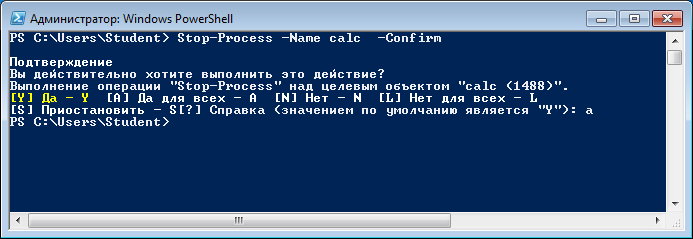


Рис. 10.

**Завершение не отвечающих процессов**

У объектов System.Diagnostics.Process, соответствующих запущенным процессам, имеется свойство Responding, которое принимает значение $False, если процесс не реагирует на запросы. Для завершения всех неотвечающих приложений нужно выделить их из общей массы с помощью командлета Where-Object, после чего остановить с помощью командлета StopProcess (рис. 11):

PS C:\Users\Student> Get-Process | Where-Object {-not $\_.Responding} | Stop-Process

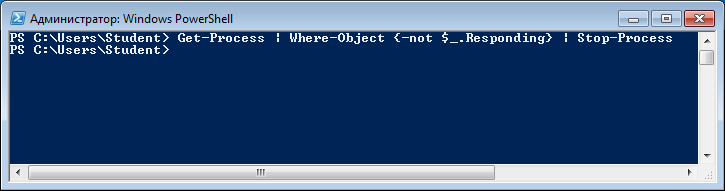


Рис. 11.

**УПРАВЛЕНИЕ СЛУЖБАМИ**

Состав имеющихся в наличии и запущенных служб зависит от версии операционной системы и установленных в ней приложений. Основным инструментом для администрирования служб в графическом режиме на локальном компьютере является консоль **Службы**, которая находится в программной группе **Администрирование** **Панели управления**.

**Просмотр списка служб**

Получить список служб, зарегистрированных на локальном компьютере, позволяет командлет Get-Service (рис. 12):

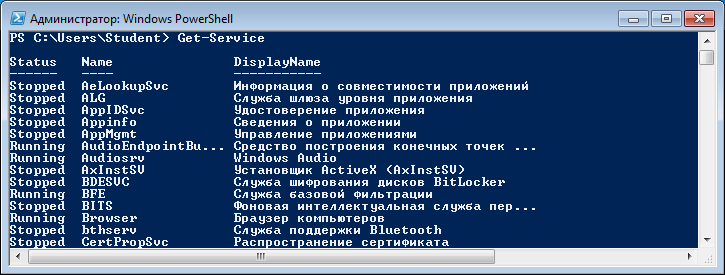


Рис. 12.

Как видим, по умолчанию отображаются имя службы (колонка Name), ее отображаемое имя (колонка DisplayName) и состояние (колонка Status).

Если нужно вывести на экран только работающие в данный момент службы, то нужно отфильтровать объекты, у которых значением свойства Status является строка "Running":

PS C:\Users\Student> Get-Service | Where-Object {$\_.Status -eq "Running"}

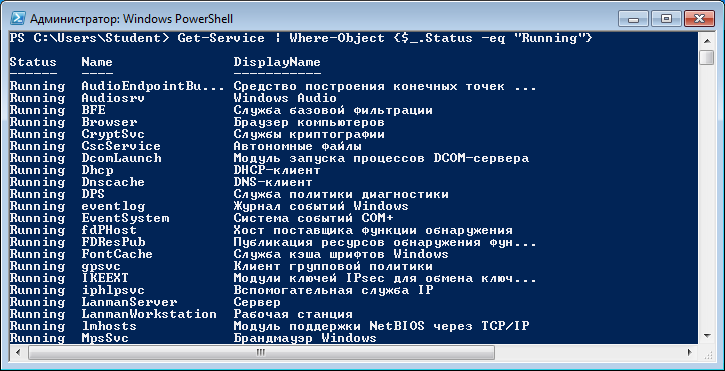


Рис. 13.

**Остановка и приостановка службы**

Локальную службу можно оставить с помощью командлета Stop-Service. Параметр -Name задает имя службы для остановки (здесь можно использовать подстановочные знаки), параметр -Force позволяет остановить указанную службу вместе со всеми службами, зависящими от нее. Например, следующая команда остановит службу Spooler (очередь печати) со всеми зависящими от нее службами:

PS C:\Users\Student> Stop-Service -Name spooler -Force

По умолчанию командлет Stop-Service не передает далее по конвейеру объекты, соответствующие останавливаемым службам, и поэтому на экране ничего не отображается. Для остановки служб с выводом информации нужно указать параметр   
-PassThru, например (рис. 14):

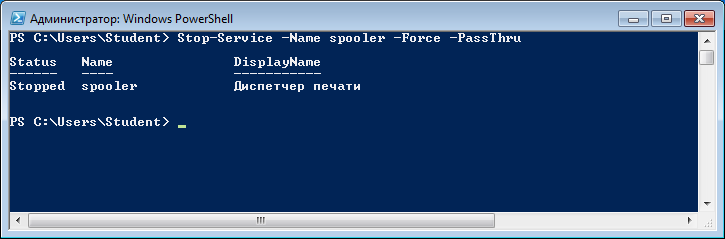


Рис. 14.

Командлет Suspend-Service позволяет приостановить работу одной или нескольких служб, имена которых задаются в качестве значения параметра -Name. При приостановке (временной остановке) службы она продолжает выполняться, однако ее действия приостанавливаются до поступления команды на возобновление работы.

Следует учитывать, что не всякую службу можно приостановить. У объектов System.ServiceProcess.ServiceController, соответствующих службам, имеется логическое свойство CanPauseAndContinue, которое равно $True, если служба может быть приостановлена. Следующая команда приостановит все службы, для которых это возможно (рис. 15):

PS C:\Users\Student> Get-Service | Where-Object {$\_.CanPauseAndContinue} | Suspend-Service -PassThru

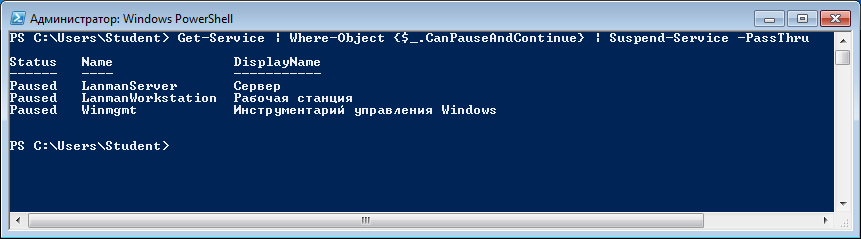


Рис. 15.

**Запуск и перезапуск служб**

Запустить службу на локальном компьютере можно с помощью командлета Start-Service. В качестве значения параметра Name указывается имя запускаемой службы. Как и в предыдущих командлетах \*-Service, после запуска службы на экран не выводится никакого сообщения; для вывода информации можно использовать параметр -PassThru. Например, следующая команда запускает службу Spooler и выводит на экран информацию об этой службе (рис. 16):

PS C:\Users\Student> Start-Service -Name spooler -PassThru

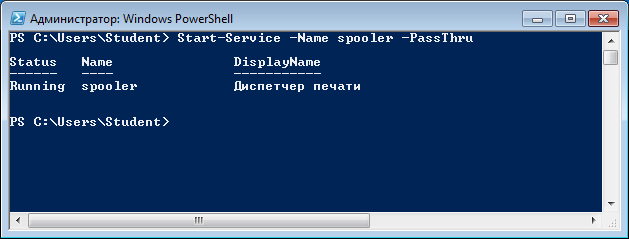


Рис. 16.

**ЧАСТЬ 2. ЗНАКОМСТВО С СИСТЕМНЫМ РЕЕСТРОМ**

Системный реестр Windows – это унифицированная база данных, содержащая информацию об аппаратной и программной конфигурации локального компьютера. Многие настройки системы возможно выполнить, лишь используя реестр, поэтому работать с ним часто бывает необходимо.

С реестром взаимодействуют ***следующие компоненты ОС:***

* *Программы установки (Windows Setup)*. Каждый раз при запуске программы Windows Setup или установочных программ для аппаратных и программных средств происходит добавление в реестр новых конфигурационных данных. Правильно разработанные программы установки считывают информацию реестра, чтобы определить, присутствуют ли в системе компоненты, обязательные для успешного завершения установки. Реестр позволяет приложениям совместно использовать конфигурационную информацию и предоставляет им больше возможностей взаимодействия между собой.
* *Распознаватель (Recognizer).* При запуске компьютера распознаватель аппаратных средств помещает в реестр список обнаруженных устройств.
* *Ядро ОС*. При старте системы ядро извлекает из реестра сведения о загружаемых драйверах устройств и порядке их загрузки. Кроме того, оно передает в реестр информацию о себе (номер версии и др.)
* *Драйверы устройств*. Драйверы обмениваются с реестром параметрами загрузки и конфигурационными данными. Каждый драйвер устройства должен сообщить об используемых им системных ресурсах, включая аппаратные прерывания.
* *Административные средства*. Эти средства, в том числе утилиты панели управления и оснастки, собранные в меню Администрирование, представляют собой удобные и безопасные (в части внесения ошибок) средства модификации реестра. Редактор реестра также полезен для просмотра и внесения изменений в конфигурацию системы.
* *Пользовательские профили (user profiles)*. ОС семейсва Windows, начиная с Windows XP/2000/NT, обеспечивают возможность создания множества пользовательских профилей. Вся информация, относящаяся к конкретному пользовательскому имени и ассоциированным с ним правам, хранится в реестре.
* *Аппаратные профили (hardware profiles)*, или профили оборудования. Реестр позволяет хранить множественные аппаратные конфигурации. Аппаратный профиль представляет собой набор инструкций, с помощью которого можно указать операционной системе, драйверы каких устройств должны загружаться при запуске компьютера.

**Структура реестра**

Логически реестр – это древовидная иерархическая база данных.

Верхний уровень образуют так называемые ветви (***Hive Keys***), которые принято обозначать аббревиатурой **HKEY\_**, где за символом подчеркивания следует название ветви.

Второй ступенью являются ***разделы*** или ***ключи*** (***Keys***). Они отображаются в программе Редактора реестра в виде подпапок ветвей **HKEY\_**.

Функционально ключи можно разделить на две условные ***категории***:

* определяемые системой (их менять нельзя);
* определяемые пользователем (эти имена могут быть изменены администратором, и такие изменения не приведут к каким-либо фатальным последствиям).

Основными в реестре являются ***5 корневых разделов***:

* **HKEY\_CLASSES\_ROOT** (**HKCR**) содержит сведения о COM-объектах и ассоциациях файлов с приложениями;
* **HKEY\_LOCAL\_MACHINE** (**HKLM**) хранит сведения о рабочей станции, драйверах и системных настройках, то есть информацию, специфичную для компьютера, а не для пользователя;
* **HKEY\_CURRENT\_CONFIG** (**HKCC**) включает в себя информацию о текущей конфигурации аппаратуры компьютера и используется в основном на компьютерах с несколькими аппаратными конфигурациями;
* **HKEY\_USERS** (**HKU**) содержит данные обо всех пользователях, зарегистрированных на рабочей станции. Здесь хранится информация о каждом пользователе (значения по умолчанию для программ, конфигурации рабочего стола и т. п.);
* **HKEY\_CURRENT\_USER** (**HKCU**) содержит настройки системы и программ, относящиеся к активному пользователю.

Ступенью ниже следуют ***подразделы*** (***Subkeys***). Их имена также могут быть определены системой или пользователем.

Последней ступенью в иерархической структуре системного реестра являются параметры (Values) – элементы реестра, содержащие саму информацию, определяющую работу ОС и компьютера в целом. Параметры представляют собой цепочку имя\_параметра – тип\_данных значение.

**Средства управления реестром**

Основным инструментом администрирования реестра Windows является программа Редактора реестра.

Кроме редактора реестра, имеются средства консоли управления. Часть программных средств, позволяющих изменять настройки реестра, расположена в Панели управления.

Практически все параметры ОС, связанные с окружением пользовательской среды, ее возможностями и ограничениями, можно изменить при помощи Редактора системных политик.

Возможно и управление реестром Windows из командной строки посредством консольного интерпретатора команд CMD или средствами командной оболочки PowerShell.

Рассмотрим редактирование реестра с использованием Редактора реестра.

Чтобы открыть окно редактора реестра, необходимо в строке поиска указать «regedit» (рис. 17).

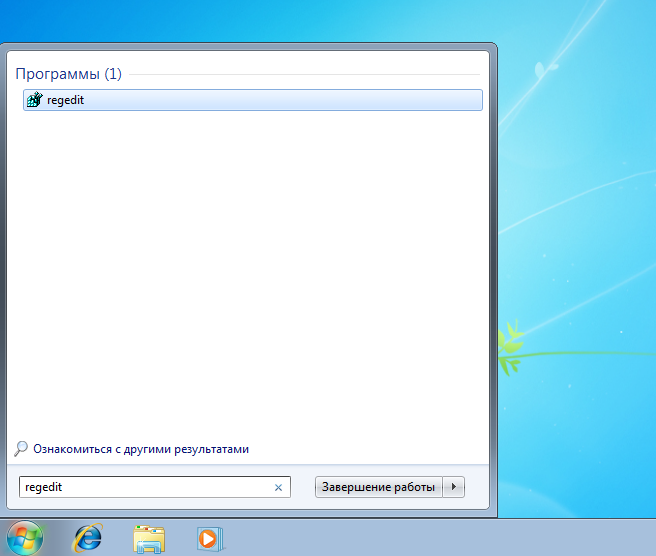


Рис. 17.

Редактор реестра в левой панели отображает структуру реестра (рис. 18).

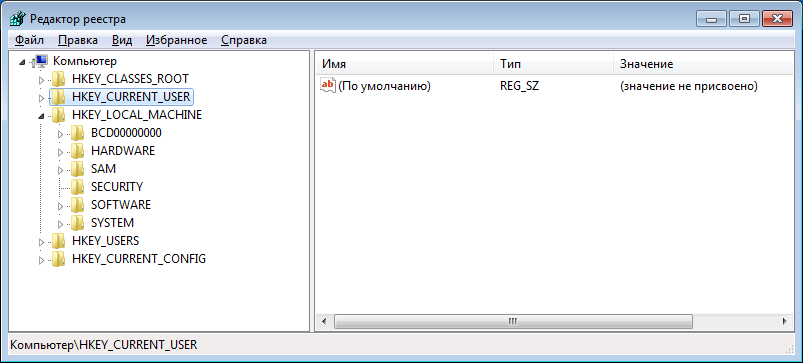


Рис. 18.

Фрагмент дерева подразделов раздела реестра **HKEY\_CURRENT\_USER** с раскрытым подразделом установленного программного обеспечения (**Software**) представлен на рис. 19.

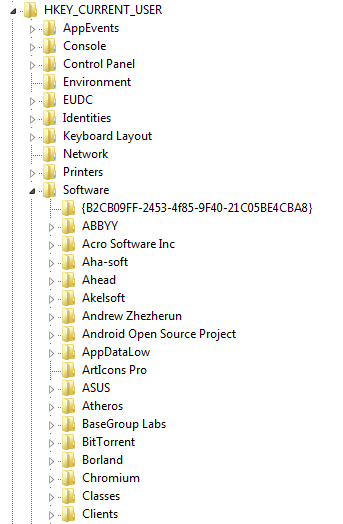


Рис. 19.

**Автозапуск программ**

Чтобы выполнить автозапуск некоторой программы при входе в учетную запись пользователя, необходимо выбрать раздел **HKEY\_CURRENT\_USER**, а в нем перейти к элементу

**HKEY\_CURRENT\_USER\Software\Microsoft\Windows\CurrentVersion\Run**

и создать строковый параметр в разделе, где имя – произвольное (должно быть информативным), а значение – запускаемая программа с необходимыми параметрами.

Последовательность действий для автозапуска Paint представлена на рис. 20.

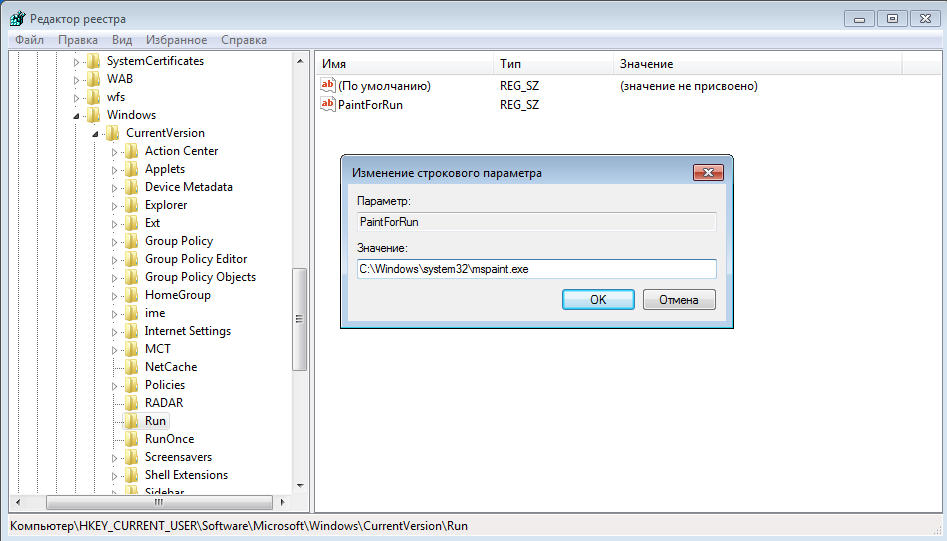


Рис. 20.

При входе в систему запустится указанная программа (в примере – Paint). Такое поведение предписано только текущему пользователю. Если необходимо реализовать такое поведение для каждого пользователя, то аналогичную ветку в разделе **HKEY\_LOCAL\_MACHINE**.

**Изменение путей к папкам для хранения основных ресурсов пользователя**

Изменить пути к папкам для хранения основных типов ресурсов пользователя, например, **Documents** – **Мои Документы**; **Music** – **Моя музыка** и т.п., осуществляется в редактированием элемента раздела

**HKEY\_CURRENT\_USER\Software\Microsoft\Windows\CurrentVersion\  
Explorer\Shell Folders**

Например, если потребуется сохранять изображения (каталог **Изображения**) в некоторой папке, например, **C:\TestFolder** (рис. 21), то необходимо редактировать указанный элемент реестра, как это представлено на рис. 22.

Изменения вступают в силу после перезагрузки компьютера.

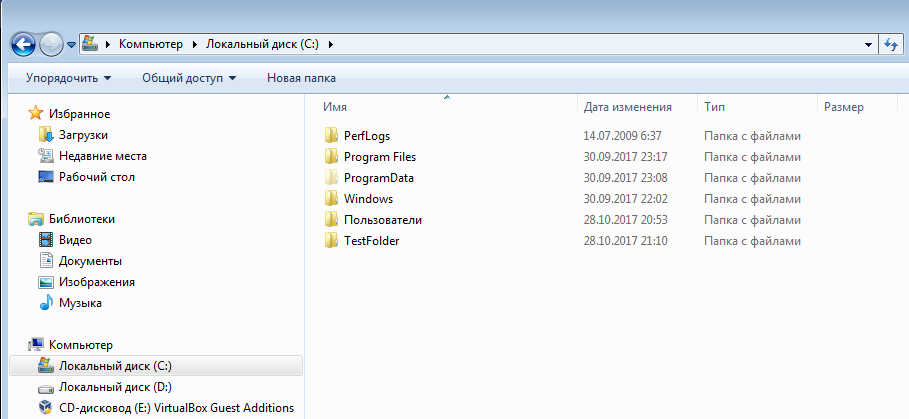


Рис. 21.

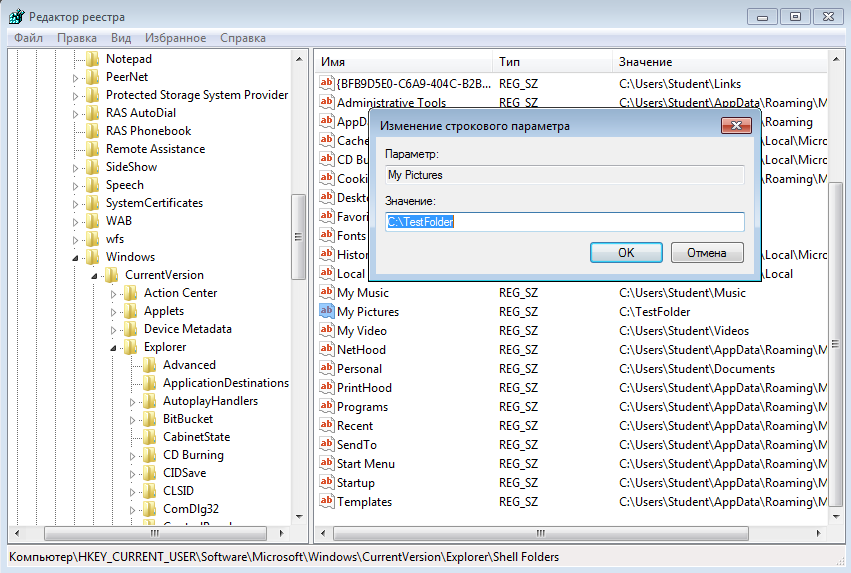


Рис. 22.

**ЧАСТЬ 3. УПРАВЛЕНИЕ ГРУППОВЫМИ ПОЛИТИКАМИ**

Эффективное функционирование ни одной многопользовательской операционной системы невозможно без четкого разграничения доступа к ресурсам. Одним из средств, позволяющих настраивать параметры безопасной работы пользователей в сети в операционных системах Windows, являются ***политики безопасности***.

Групповая политика представляет собой самый простой способ настройки компьютера и параметров пользователей в сетях на основе доменных служб Active Directory.

Настройки групповых политик находятся в ***объектах групповых политик*** (Group Policy Object, GPO).

Политики безопасности хранятся в двух ***типах объектов GPO***:

* локальном объекте групповой политики;
* объекте групповой политики домена.

После создания GPO ассоциируется с определенным контейнером, и в результате групповые политики, хранящиеся в данном GPO, будут выполняться для всех компьютеров и пользователей, находящихся в этом контейнере. Дополнительными средствами настройки групповых политик в контейнере являются группы безопасности и дискреционные разрешения доступа.

GPO создается с помощью оснастки консоли управления **Групповая политика**.

Для управления групповыми политиками необходимо запустить оснастку gpedit.msc (рис. 23).

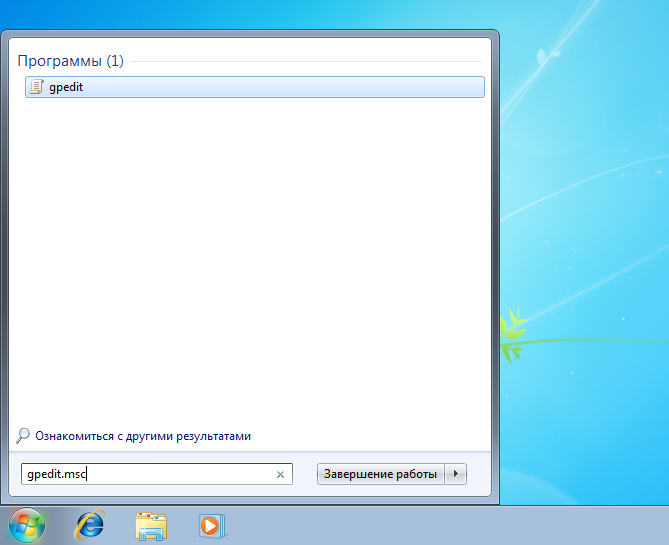


Рис. 23

После запуска оснастки **Групповая политика** в окне структуры появляется набор узлов, которые являются расширениями этой оснастки. Корневой узел представляет собой GPO, присоединенный к контейнеру. Затем пространство имен подразделяется на два узла более низкого уровня (рис. 24):

* **Конфигурация компьютера** (**Computer Configuration**);
* **Конфигурация пользователя** (**User Configuration**).

Используя их, можно создавать и настраивать групповые политики для компьютера и пользователей.

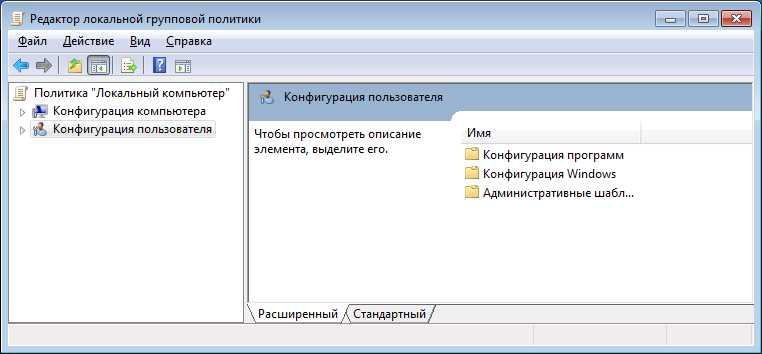


Рис. 24.

Узел **Конфигурация компьютера** содержит параметры всех политик, определяющих работу компьютера. Они регулируют функционирование операционной системы, вид рабочего стола, задают параметры выполняемых приложений, определяют работу средств обеспечения безопасности и т. д. Групповая политика применяется к компьютеру на этапе загрузки системы и в дальнейшем при выполнении циклов обновления.

Узел **Конфигурация пользователя** содержит параметры всех политик, определяющих работу пользователя на компьютере. Они регулируют вид рабочего стола, как и в предыдущем случае, задают параметры выполняющихся приложений, определяют работу средств обеспечения безопасности и пользовательских сценариев входа и выхода. Групповая политика применяется к пользователю при его регистрации на компьютере и в дальнейшем при выполнении циклов обновления.

Ниже родительских узлов **Конфигурация компьютера** и **Конфигурация пользователя** находятся дочерние узлы, каждый из которых является полноценным расширением оснастки **Групповая политика** (рис. 25). Они могут находиться в обоих родительских узлах, хотя и с различными параметрами, или индивидуально расширять узлы **Конфигурация компьютера** или **Конфигурация пользователя**.

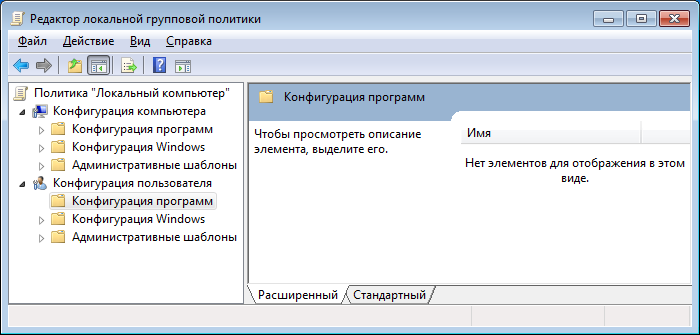


Рис. 25.

С помощью расширения **Административные шаблоны** администратор системы может настроить целый набор параметров реестра, задающих режим функционирования компонентов операционной системы и приложений.

Задать конкретные параметры, доступные для модификации с помощью интерфейса пользователя оснастки **Групповая политика**, можно с помощью специальных административных шаблонов. Модифицируемые значения параметров реестра, относящиеся к зарегистрированному в компьютере пользователю, записываются в раздел реестра **HKEY\_CURRENT\_USER**. Значения шаблонов, относящиеся к компьютеру, записываются в раздел **HKEY\_LOCAL\_MACHINE**.

Групповые политики имеют более высокий приоритет по сравнению с реестром.

С оснастками, как таковыми, мы уже встречались, отключая или запуская указанную службу (лабораторная работа №2).

Файлы \*.msc, являющиеся оснастками, скрываются системой (по умолчанию Проводником не отображаются). Системные оснастки расположены в папке **C:\Windows\System32**.

Окно редактора локальной групповой политики с раскрытыми узлами **Административные шаблоны** и **Система** представлено на рис. 26.

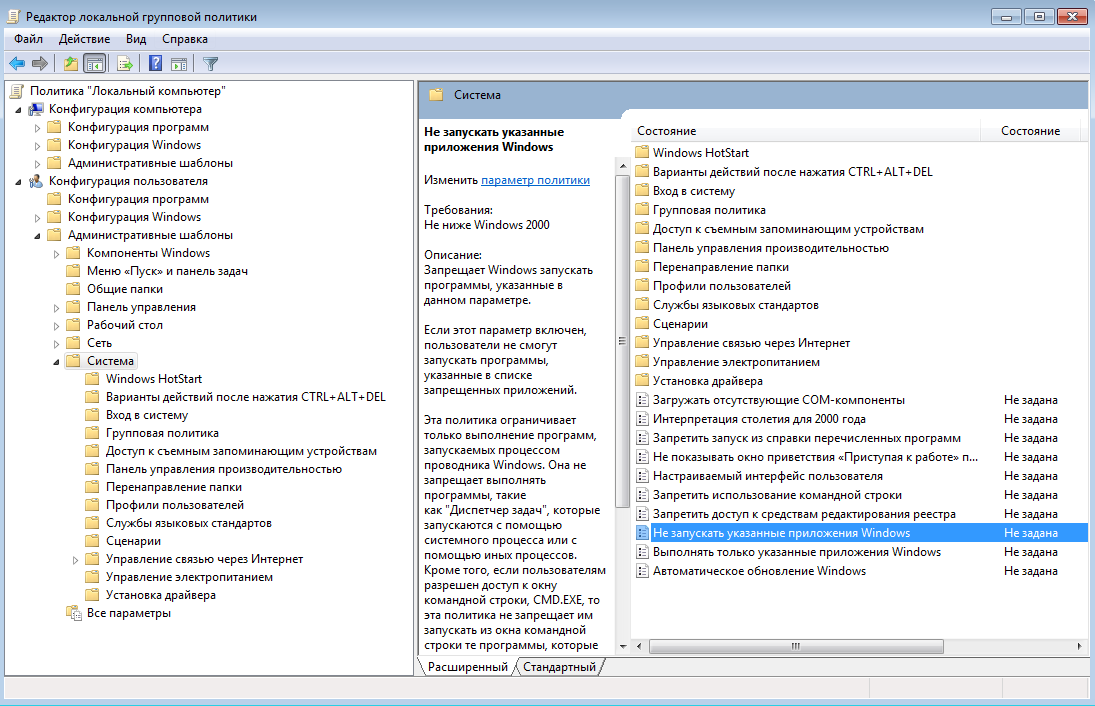


Рис. 26

Принцип работы с редактором понятен на интуитивном уровне. Требуется выбрать соответствующий раздел в представленной в левой панели иерархии и осуществить редактирование соответствующего элемента в правой панели. При выделении интересующего элемента в правой панели по нему отображается справка.

Так, например, чтобы запретить запуск определенных программ, необходимо выбрать узел

**Конфигурация пользователя\Административные шаблоны\Система**

и в правом окне запустить выбрать *Не запускать указанные приложения Windows*. Двойной щелчок мыши открывает окно редактирования, где можно поставить переключатель в нужное положение (*Включен (Включить)*).

На рис. 27 представлен вариант редактирования соответствующего свойства для Paint. После перезагрузки Paint станет недоступным, о чем система сообщит при попытке запустить запрещенное приложение (рис. 28).

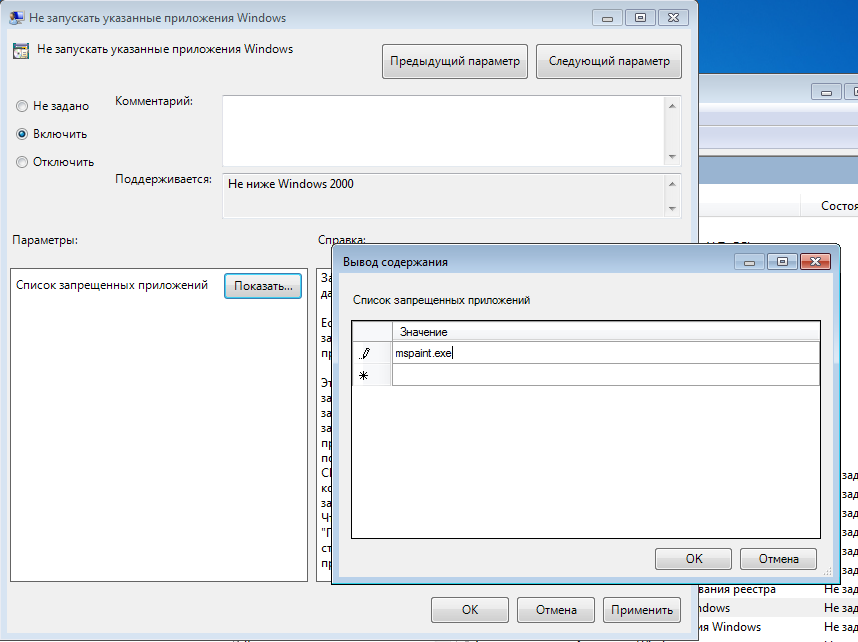


Рис. 27

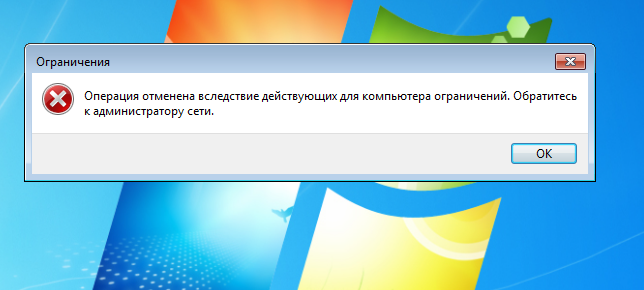


Рис. 28