Using Sc.exe to Develop Windows 2000 Services

The command-line tool Sc.exe can help you develop your services for Microsoft® Windows® 2000. Sc.exe, which is provided in the Microsoft® Windows® 2000 Resource Kits, implements calls to all of the Windows 2000 service control application programming interface (API) functions. You can set the parameters to these functions by specifying them on the command line. Sc.exe also displays service status and retrieves the values stored in the status structure fields. The tool also lets you specify the name of a remote computer so that you can call the service API functions or view the service status structures on the remote computer.

This document describes how to use the Sc.exe tool to obtain detailed information about Windows 2000 services. For more information about developing Windows 2000 services and for detailed reference information about the Windows 2000 Service API functions, such as ControlService and QueryServiceStatus, see the Win32 SDK documentation (Microsoft Development Library, Product Documentation, SDKs).

Sc.exe is a development tool that provides more detailed and accurate information about services than the two end-user tools that are provided with the operating system. The Services application in Control Panel and the network command-line interface, Net.exe, can tell you that a service is running, stopped, or paused. These tools are fine for completely debugged and running services when everything is going smoothly. But when things go wrong, as sometimes happens when developing new code, the information provided by these tools can be misleading.

For instance, if during the development stage your service stops responding, or “hangs,” in the start-pending state, Services in Control Panel and Net.exe report its state as running. If the service stops responding when in the stop-pending state, NET START reports the status as running, Services reports it as stopped. If you then attempt to start it, Services tells you the service is already running. This can be quite confusing.

Sc.exe lets you query the service status and retrieve the values stored in the status structure fields. The Net.exe program and Services don't provide the complete status for the service. The SC program, however, will tell you the exact state of the service as well as show you the last checkpoint number and wait hint. The checkpoint can then be used as a debugging tool because it provides a clear indication of how far along the initialization had progressed before the program froze.

Sc.exe also allows you to call any of the service control API functions and vary any of the parameters from the command line. This offers several advantages to the service developer. For instance, it provides a convenient way of creating or configuring the service information in the registry and the Service Control Manager's database. The developer doesn't have to configure the service by manually creating entries in the registry and then rebooting the computer in order to force the Service Control Manager to update its database.

Caution: do not modify the Services key of the registry directly, as this could cause unpredictable behavior in starting, stopping, or controlling the service.

As a command line program, Sc.exe can also be used to create tests for your service. You can create batch (command) files that call Sc.exe with various parameters that control the service. This is useful if you want to see how your service behaves when it is repeatedly started and stopped. If you have more than one service in your service process, you can leave one service running so that the process doesn't go away, and then repeatedly start and stop the other service while looking for evidence of memory leaks due to an incomplete cleanup.

The following sections contain reference information for the commands **SC**, **SC QC**, and **SC QUERY**.

SC

The SC command-line tool uses the following syntax:

Syntax1
sc [Servername] Command Servicename [Optionname= Optionvalue...]

Syntax2
sc [Command]

Use Syntax1 to run Sc.exe. Use Syntax2 to display help information (except for the **query** command--see the "Comments" section later in this document for more information).

Parameters

Servername
Optional. Specifies the name of the server when you want to run the commands on a remote computer. The name must start with two double backslash characters, such as "\\myserver". To run SC on the local computer, do not supply this parameter.

Command
Specifies the SC command. Note that many of the SC commands require administrative privilege on the specified computer. SC supports the following commands:

Command	Description
Config	Changes the configuration of a service (persistent).
Continue	Sends a CONTINUE control request to a service.
Control	Sends a control to a service.
Create	Creates a service (adds it to the registry).
Delete	Deletes a service (from the registry).
Description	Changes the description of a service.
EnumDepend	Enumerates services that depend on the given service.
Failure	Changes the actions taken by a service upon failure.
GetDisplayName	Gets the DisplayName for a service.
GetKeyName	Gets the ServiceKeyName for a service.
Interrogate	Sends an INTERROGATE control request to a service.
Pause	Sends a PAUSE control request to a service.
Qc	Queries configuration for the service. For detailed information, see the reference section "SC QC" later in this document.
Qdescription	Queries the description of a service.

Qfailure Queries the actions taken by a service upon failure.

Query Queries the status for a service, or enumerates the status for types of services. For detailed information, see the reference section, "SC QUERY."

QueryEx Queries the status and extended information for a service, or enumerates the status and extended information for types of services.

SdShow Displays a service's security descriptor using SDDL.

SdSet Sets a service's security descriptor using SDDL.

Start Starts a service.

Stop Sends a STOP request to a service.

Servicename
Specifies the name given to the service key in the registry. Note that this is different from the display name, which is what you see with NET START and the Services. SC uses the service key name as the primary identifier for the service.

Optionname
The Optionname and Optionvalue parameters allow you to specify the names and values of optional command parameters. Note that there is no space between the Optionname and the equal sign. You can supply zero, one, or more optional parameter name and value pairs.

For a list of the available Optionname values, request help for the command by entering **sc Command**, where Command is a valid SC command name, as listed in the Command parameter description above.

Optionvalue
Specifies the value for the parameter named by Optionname. The range of valid values is often restricted for each Optionname. For a list of available values for a command, request help for that specific command.

Comments

Many of the commands require administrator privileges, so it's a good idea to make sure that you are an administrator of the computer where the development is being done.

When you enter **SC** with no parameters, Sc.exe displays help information that lists the available commands. When you enter **SC** followed by a command name, you can get specific information about that command. For example, enter **SC CREATE** to get help specific to the **CREATE** command.

The exception to this syntax is **SC QUERY**, which dumps the status of all services and drivers currently running in the system. For help on the **QUERY** command, enter **SC**, and respond to the prompt, "Would you like to see help for the QUERY command?" by entering **Y** for yes.

When you use the **START** command, you can pass arguments to the service's ServiceMain function. Note that the arguments are not passed to the service process's main function.

SC CREATE

The **SC CREATE** command creates an entry for the service in the registry and in the Service Control Manager's database.

Syntax1
sc [Servername] create Servicename [Optionname=Optionvalue...]

Parameters

Servername
Optional. Specifies the name of the server when you want to run the commands on a remote computer. The name must start with two double backslash characters, such as "\\myserver". To run SC on the local computer, do not supply this parameter.

Servicename
Specifies the name given to the service key in the registry. Note that this is different from the display name, which is what you see with NET START and Services. SC uses the service key name as the primary identifier for the service.

Optionname
The Optionname and Optionvalue parameters allow you to specify the names and values of optional parameters. Note that there is no space between the Optionname and the equal sign. You can supply zero, one, or more optional parameters name and value pairs. The SC QUERY command supports the following values:

Optionname Optionvalue
Description

type= own, share, interact, kernel, filesys
Type of service being created. Optionvalues include types used by drivers. (default = own)

start= boot, system, auto, demand, disabled
Start type for the service. Option values include types used by drivers. (default = demand)

error= normal, severe, critical, ignore
Severity of error if the service fails to start during boot. (default = normal)

binPath= (string)
Path name to the service binary file. There is no default for this. This string must be supplied.

group= (string)
Name of group of which this service is a member. The list of groups is stored in the registry under the HKLM\System\CurrentControlSet\Control\ServiceGroupOrder key. (default = nothing)

tag= (string)
If this string is set to "yes", SC will obtain a TagId from the CreateService call. Tags are only used for boot- and system-start drivers. (default = nothing)

depend= (strings separated by forward-slash)
Names of services or groups which must start before this service.

obj= (string)
Name of account in which the service will run. For drivers, this is the Windows 2000 driver object name. (default = LocalSystem)

DisplayName= (string)
A friendly name that can be used by user-interface programs to identify the service.

password= (string)
A password string. This is required if an account other than LocalSystem is used.

Optionvalue
Specifies the value for the parameter named by Optionname. See the Optionname reference for a list of supported values. When a string is to be input, the use of empty quotes means that an empty string will be passed in.

Comments

The **SC CREATE** command calls the CreateService API function.

Example One

The following example creates a registry entry for the service named "NewService" on the computer called "\\myserver":

sc \\myserver create NewService binpath= c:\nt\system32\NewServ.exe

By default this service will be created as a WIN32_OWN_PROCESS with a SERVICE_DEMAND_START start-type. It will not have any dependencies, and will run in the LocalSystem security context.

Example Two

The following example creates the service on the local computer as an auto-start service that runs in a shared process. It has dependencies on the TDI group and on the NetBIOS service. Notice that you must add quotes around the list of space-separated dependencies.

sc create NewService binpath= c:\nt\system32\NewServ.exe type= share
start= auto depend= "+TDI Netbios"

Example Three

The service developer can run the service in the context of the kernel debugger by temporarily changing the binary path (image path) for the service. The following example shows how to call SC to change the service configuration:

sc config NewService binpath= "ntsd -d c:\nt\system32\NewServ.exe"

This example causes the Service Control Manager to invoke Ntsd.exe with the following argument string:

"-d c:\nt\system32\NewServ.exe".

NTSD will in turn break into the debugger when it loads Newserv.exe so that breakpoints can be set in the service code. This is especially useful for debugging problems in the service's start-up code.

SC QC

The SC QC **query configuration** command lists information about the service configuration from the QUERY_SERVICE_CONFIG structure.

Syntax1
sc [Servername] qc Servicename [Buffersize]

Parameters

Servername
Optional. Specifies the name of the server when you want to run the commands on a remote computer. The name must start with two double backslash characters, such as "\\myserver". To run SC on the local computer, do not supply this parameter.

Servicename
Specifies the name given to the service key in the registry. Note that this is different from the display name, which is what you see with NET START and the Services application in Control Panel. SC uses the service key name as the primary identifier for the service.

Buffersize
Optional. Specifies the size of the buffer passed to the QueryServiceConfig API in bytes.

Comments

The SC QC command displays the contents of the QUERY_SERVICE_CONFIG structure:

The following table displays information displayed by the SC QC command, followed by the corresponding field from the QUERY_SERVICE_CONFIG structure.

TYPE	dwServiceType
START_TYPE	dwStartType
ERROR_CONTROL	dwErrorControl
BINARY_PATH_NAME	lpBinaryPathName
LOAD_ORDER_GROUP	lpLoadOrderGroup
TAG	dwTagId
DISPLAY_NAME	lpDisplayName
DEPENDENCIES	lpDependencies
SERVICE_START_NAME	lpServiceStartName

Example One

The following example queries the configuration of the service named "NewService" that was created in Example One for the **Create** command:

sc \\myserver qc NewService

The SC tool displays the following information:

```
SERVICE_NAME: NewService
      TYPE               : 20  WIN32_OWN_PROCESS
      START_TYPE         : 3   DEMAND_START
```

```
ERROR_CONTROL      : 1   NORMAL
BINARY_PATH_NAME   : c:\nt\system32\NewServ.exe
LOAD_ORDER_GROUP   :
TAG                : 0
DISPLAY_NAME       : NewService
DEPENDENCIES       :
SERVICE_START_NAME : LocalSystem
```

NewService runs in its own process. It will not be auto-started. The binary file name is c:\nt\system32\ Newserv.exe. This service doesn't depend on any other services, and will run in the Local System security context. Because this command displays the results from a call to QueryServiceStatus, a more detailed explanation of these results can be obtained from the SDK documentation on that API function.

SC QUERY

The SC QUERY command obtains information about the service.

Syntax
sc [Servername] query { Servicename | Optionname= Optionvalue... }

Parameters

Servername
Optional. Specifies the name of the server when you want to run the commands on a remote computer. The name must start with two double backslash characters, such as "\\myserver". To run SC on the local computer, do not supply this parameter.

Servicename
Specifies the name given to the service key in the registry. Note that this is different from the display name, which is what you see with NET START and the Services Control Panel application. SC uses the service key name as the primary identifier for the service. Note that you can supply either the Servicename or the options, but not both. When the Servicename is supplied, the other options are ignored. When the options are supplied, SC lists all the services that fit the given characteristics.

Optionname
The Optionname and Optionvalue parameters allow you to specify the names and values of optional parameters. Note that there is no space between the Optionname and the equal sign. You can supply zero, one, or more optional parameters name and value pairs. The SC QUERY command supports the following values:

Optionname Optionvalue
Description

type= driver, service, all
Type of services to enumerate (default = service)

state= active, inactive, all
State of services to enumerate (default = active)

bufsize= (numeric value)
The size in bytes of the enumeration buffer (default = 1024 bytes)

ri= (numeric value)
The index number at which to begin/resume the enumeration (default = 0)

Optionvalue
Specifies the value for the parameter named by Optionname. See the Optionname reference for a list of supported values.

Comments

The SC QUERY command displays the contents of the SERVICE_STATUS structure:

The following table shows information displayed by **SC QUERY** command, followed by the corresponding field from the SERVICE_STATUS structure

TYPE	dwServiceType
STATE	dwCurrentState, dwControlsAccepted
WIN32_EXIT_CODE	dwWin32ExitCode
SERVICE_EXIT_CODE	dwServiceSpecificExitCode
CHECKPOINT	dwCheckPoint
WAIT_HINT	dwWaitHint

Using the **SC QUERY** command after starting the computer will tell you whether or not an attempt was made to start this service. If the service was started successfully, the WIN32_EXIT_CODE field should contain a zero (0). If the service failed to start when an attempt was made, this field should contain an exit code provided by the service when it realized it couldn't start.

Examples

To query the status of the service, "NewService," enter:

sc query NewService

SC displays the following information:

```
SERVICE_NAME: NewService
    TYPE               : 20  WIN32_OWN_PROCESS
    STATE              : 1  STOPPED
                          (NOT_STOPPABLE,NOT_PAUSABLE,IGNORES_SHUTDOWN)
    WIN32_EXIT_CODE    : 1077       (0x435)
    SERVICE_EXIT_CODE  : 0  (0x0)
    CHECKPOINT         : 0x0
    WAIT_HINT          : 0x0
```

Notice that there is an exit code for this service, even though it has not yet been run. Entering the Windows 2000 command **net helpmsg 1077** at the command line returns the following text information for error 1077:

No attempts to start the service have been made since the last boot.

Net Helpmsg can be used to display the text for most Windows 2000 error messages. This particular exit code indicates that this service hasn't yet been run. Although obvious in this case, this particular exit code is a useful one to look for if you are expecting your service to be auto-started or perhaps when another auto-start service has a dependency on your service.

Here are some further examples of SC QUERY usage:

To enumerate status for active services and drivers, use the following command:

sc query

To display status for the messenger service, use the following command:

sc query messenger

To enumerate only active drivers, use the following command:

sc query type= driver

To enumerate only Win32 services, use the following command:

sc query type= service

To enumerate all services and drivers, use the following command:

sc query state= all

To enumerate with a 50 byte buffer, use the following command:

sc query bufsize= 50

To enumerate with resume index = 14, use the following command:

sc query ri= 14

To enumerate all interactive services, use the following command:

sc query type= service type= interact