To Main Page

View in Russian

Writing NT service using VB6/VB5

It is known, that Visual Basic isn't most appropriate tool for developing of Windows NT/2000/XP services. The problem is, for service development is necessary to use API function **CreateThread**, which is not supported nether in VB5, nor in VB6. VB allows creation of multi-thread programs, but not using **CreateThread** function. VB5 utilizes **Err** object, common for all threads created by **CreateThread** function. It is inadmissible, because values of this object may be changed from different threads in unpredictable order, and ever simultaneously on multi-processor computers. On the contrary, VB6 uses Thread Local Storage (TLS) for **Err** object (and not only for it), but TLS isn't initialized when **CreateThread** function creates new thread, and therefore, after simple recompilation of the program in VB6 it doesn't work at all (rather, it works only being compiled in p-code with common **Err** object).

Matthew Curland suggested rather complicated method of threads creation in VB (see links), but for purpose of service development more simple solution may be applied. The program will work without TLS, if in new thread used only arithmetic operators and API functions, declared in type library, and **Err** object isn't used at all. It is well known that the properties of **Err** object became changed in two cases: when occurs Run-Time error (**Err.Number**) and after every API call (**Err.LastDllError**). The first may be avoided by writing error-free code, the second - by declaring API calls in Type Library (API functions declarations must not contain **usesgetlasterror** attributes, therefore type libraries from the popular Bruce McKinney's book are not suitable). Functional part of the service must work in main thread, and above limitations doesn't apply to it. This approach must work both in VB6 and VB5.

Of course, you can use free NTSVC.OCX control from Microsoft to create your service, and this way is simple and reliable, but it has one disadvantage: you can't set "Unattended execution" option for it. You can't use any OCX controls when "Unattended execution" is set while this mode is preferable for service. In addition, because this control distributed in form of source code, there are some incompatible compiled versions of it.

This sample is written using VB6 without any external components. As service is compiled for unattended execution, it has no visual interface. Use Event Viewer to read messages written by the service to the Application Log.

The functional part of service (which is absent in this sample) must be driven by the events from Service Dispatcher. All events must be processed during few seconds, otherwise the service will be unable to respond on requests from Service Dispatcher.

See also:

Microsoft Knowledge Base Q137890, Q170883, Q175948,

An OLE Control for Creating Win32 Services in Visual Basic,

Manipulate Windows NT Services by Writing a Service Control Program,

Creating an Agent NT Service with VB,

How-To Run Your Application as a Service,

How-To Run Your Application as a Service - Part II,

Matthew Curland's article "Create Worker Threads in DLLs" in June/1999 issue of "Visual Basic Programmer's Journal".

Updates of source code:

06 June 2004

1. All API functions calls except GetVersionEx changed to Unicode versions, in code and in type library. Added new Enum members to type library to support new Windows 2000 control codes.

2. Added MsgWaitObj function to prevent blocking of messages processing. All calls of WaitForSingleObject and WaitForMultipleObjects in Sub Main replaced with MsgWaitObj.

Questions and Answers:

Q: Is there any way to make the events in Event Viewer show up as the APP name instead of VBRuntime?

A: Of course, yes. For example, see the solution in this article. The code for the article you can download here.

Q: I have run the ServiceSampleControl.vbp and installed the service, and then tried starting the service but it didn't not start. When running the ServiceSample.vbp the error states that it must be started as a service.

A: Did you try starting SvSampleControl.exe? By this sample design, the service program (SvSample.exe) must be in the same directory as the control program (SvSampleControl.exe) or its project (ServiceSampleControl.vbp), if you prefer starting it in IDE. Because the project of the control program located in the separate directory, the service executable was not found. See System event log - this fact must be reflected there. Of course, this behaviour may be changed by rewriting some bytes in the control program.

Q: Is it possible in to turn on "Allow service to interact with Desktop"?.

A: Yes, it is possible. When you call CreateService or ChangeServiceConfig functions, pass this value to dwServiceType parameter:

```
SERVICE_WIN32_OWN_PROCESS Or SERVICE_INTERACTIVE_PROCESS

Const SERVICE_WIN32_OWN_PROCESS = &h10&
Const SERVICE_INTERACTIVE_PROCESS = &h100&
```

Q: Is it possible to start a service as another user (administrator) with "Interact with Desktop"?

A: No, it is impossible, only LocalSystem services can interact with desktop. It is security feature.
Usually in this situation programmers divide an application into two parts: the service running under some user account with sufficient rights provides main work and network communication; and ordinary auto-start executable provides interaction with current user. These two parts interact each other using one of known methods: COM, named pipes, shared memory etc. Using any of these methods is not simple in conformity with services because they must be modified to overcome security barriers.

Q: Have you noticed any problems with using your service with local DDE application? When that service is running, local DDE application (and some installs written with InstallShield) hang. The process exists on the machine, but the exe does not run. As soon as I issue a STOP to the serivce, the local DDE application begins.

A: Try to replace all wait functions (WaitForSingleObject and others) with my non-blocking equivalent function MsgWaitObj. Using blocking versions of wait functions in VB program is not correct even if the program have no visible interface (don't touch WaitForSingleObject in NTService.bas: using this function in service thread is lawful, but DoEvents from MsgWaitObj may be not thread-safe, and used there API functions are not declared in type library).

Note: updated source code of service sample is available for downloading, it takes into account above-mentioned considerations.

Q: I want to customize your "NT Service module" application for running my .exe as a NT service. Is there anything I should worry about during customization?

A: The first rule for service: it must respond to Stop request as soon as possible. If your program performs long operations, insert intermediate checks for hStopPendingEvent event:

```
If WaitForSingleObject(hStopPendingEvent, 0&) = 0& Then ... cancel operation
```

The next rule: initialization code must be brief. That is the program must execute

```
SetServiceState SERVICE_RUNNING
```

line within 1-2 seconds.

The sample introduces two different approaches in service design.
1. The program uses objects which respond to external events (for example, you are writing Internet server, which responds to data arrival in socket). You have initialization code, termination code and no code in main loop. The interval for main loop you may set to INFINITE.
2. Your program performs periodic tasks (for example, you are writing time scheduler). You have no initialization code (or very brief assignment initial values for variables at that place), no termination code and all functional code in main

loop.
Of course, you may combine these approaches.

The next thing is security. If the service runs in LocalSystem account, it can't access network shares at all, if it runs in some User account, it can access secure objects in accordance with User rights. If your service is COM server, you must use DCOM to access it from user's application even from the same computer. You must add and modify some registry settings to make COM service work properly.

Q: I try to write some value to the registry (HKCU\...) from the service, but registry editor states that nothing was written.
A: Your problem is, the service runs in context of another account (typically, "LocalSystem"). That is, HKCU for your interactive session and for service point to different registry hives. Use "HKLM\Software\Your Company\Your Program" key to store data instead. Or start your service in context of your account (it is adjustable in Control Panel\Services).

Download source code
vb6svc.zip (34 kB, English interface)

This page has been updated last time on **06-Jun-2004**

Write me: sm@smsoft.ru