如何用C语言做Windows服务开机

**创建一个Windows服务可以通过C语言实现，步骤包括：编写服务的主函数、注册服务、实现服务控制处理程序等。本文将详细介绍如何用C语言创建和管理Windows服务，重点包括**编写服务的主函数、注册服务、实现服务控制处理程序、调试和部署服务。下面将对编写服务的主函数进行详细介绍。

**编写服务的主函数是创建Windows服务的关键步骤之一。服务主函数是服务的核心逻辑所在，它负责初始化服务并启动服务的主要工作线程。**

# 一、编写服务的主函数

编写服务的主函数是创建Windows服务的核心步骤，它包括初始化服务并启动服务的主要工作线程。以下是详细步骤：

## 定义服务主函数

服务主函数是服务的入口点，通常命名为ServiceMain。在这个函数中，我们需要进行初始化操作并启动服务的主要工作线程。

void WINAPI ServiceMain(DWORD dwArgc, LPTSTR \*lpszArgv) {

// 初始化服务状态和服务句柄

g\_ServiceStatusHandle = RegisterServiceCtrlHandler(SERVICE\_NAME, ServiceCtrlHandler);

if (g\_ServiceStatusHandle == NULL) {

return;

}

// 初始化服务状态

g\_ServiceStatus.dwServiceType = SERVICE\_WIN32\_OWN\_PROCESS;

g\_ServiceStatus.dwCurrentState = SERVICE\_START\_PENDING;

g\_ServiceStatus.dwControlsAccepted = SERVICE\_ACCEPT\_STOP;

g\_ServiceStatus.dwWin32ExitCode = 0;

g\_ServiceStatus.dwServiceSpecificExitCode = 0;

g\_ServiceStatus.dwCheckPoint = 0;

g\_ServiceStatus.dwWaitHint = 0;

// 通知服务控制管理器服务正在启动

SetServiceStatus(g\_ServiceStatusHandle, &g\_ServiceStatus);

// 启动服务的主要工作线程

ServiceWorkerThread();

}

## 实现服务控制处理程序

服务控制处理程序用于处理来自服务控制管理器的控制请求，如启动、停止等。通常命名为ServiceCtrlHandler。

void WINAPI ServiceCtrlHandler(DWORD dwCtrl) {

switch (dwCtrl) {

case SERVICE\_CONTROL\_STOP:

// 处理停止请求

g\_ServiceStatus.dwCurrentState = SERVICE\_STOP\_PENDING;

SetServiceStatus(g\_ServiceStatusHandle, &g\_ServiceStatus);

// 通知主线程停止服务

SetEvent(g\_ServiceStopEvent);

break;

default:

break;

}

}

## 启动服务工作线程

服务的主要工作通常在一个单独的线程中进行，以确保服务的响应性。可以使用CreateThread函数创建一个工作线程。

void ServiceWorkerThread() {

// 初始化服务工作线程

while (WaitForSingleObject(g\_ServiceStopEvent, 0) != WAIT\_OBJECT\_0) {

// 这里是服务的主要工作逻辑

// 例如，定期执行某些任务

Sleep(1000); // 模拟工作

}

// 设置服务状态为已停止

g\_ServiceStatus.dwCurrentState = SERVICE\_STOPPED;

SetServiceStatus(g\_ServiceStatusHandle, &g\_ServiceStatus);

}

## 注册服务

使用CreateService函数可以将服务注册到服务控制管理器中。这个过程通常在安装服务的过程中完成。

SC\_HANDLE schSCManager = OpenSCManager(NULL, NULL, SC\_MANAGER\_CREATE\_SERVICE);

if (schSCManager == NULL) {

// 打开服务控制管理器失败

return;

}

SC\_HANDLE schService = CreateService(

schSCManager, // 服务控制管理器句柄

SERVICE\_NAME, // 服务名称

DISPLAY\_NAME, // 显示名称

SERVICE\_ALL\_ACCESS, // 访问权限

SERVICE\_WIN32\_OWN\_PROCESS, // 服务类型

SERVICE\_DEMAND\_START, // 启动类型

SERVICE\_ERROR\_NORMAL, // 错误控制类型

SERVICE\_BINARY\_PATH\_NAME, // 服务可执行文件路径

NULL, // 加载顺序组

NULL, // 服务标识

NULL, // 依赖项

NULL, // 服务启动帐户

NULL // 服务启动密码

);

if (schService == NULL) {

// 创建服务失败

CloseServiceHandle(schSCManager);

return;

}

// 成功创建服务

CloseServiceHandle(schService);

CloseServiceHandle(schSCManager);

# 二、服务的安装与启动

## 安装服务

在服务安装阶段，需要将服务注册到系统中。可以使用一个简单的安装程序，或者通过命令行工具来完成服务的安装。以下是一个简单的安装程序示例：

void InstallService() {

SC\_HANDLE schSCManager = OpenSCManager(NULL, NULL, SC\_MANAGER\_CREATE\_SERVICE);

if (schSCManager == NULL) {

// 打开服务控制管理器失败

return;

}

SC\_HANDLE schService = CreateService(

schSCManager, // 服务控制管理器句柄

SERVICE\_NAME, // 服务名称

DISPLAY\_NAME, // 显示名称

SERVICE\_ALL\_ACCESS, // 访问权限

SERVICE\_WIN32\_OWN\_PROCESS, // 服务类型

SERVICE\_DEMAND\_START, // 启动类型

SERVICE\_ERROR\_NORMAL, // 错误控制类型

SERVICE\_BINARY\_PATH\_NAME, // 服务可执行文件路径

NULL, // 加载顺序组

NULL, // 服务标识

NULL, // 依赖项

NULL, // 服务启动帐户

NULL // 服务启动密码

);

if (schService == NULL) {

// 创建服务失败

CloseServiceHandle(schSCManager);

return;

}

// 成功创建服务

CloseServiceHandle(schService);

CloseServiceHandle(schSCManager);

}

## 启动服务

安装完成后，可以使用StartService函数启动服务。

void StartMyService() {

SC\_HANDLE schSCManager = OpenSCManager(NULL, NULL, SC\_MANAGER\_CONNECT);

if (schSCManager == NULL) {

// 打开服务控制管理器失败

return;

}

SC\_HANDLE schService = OpenService(schSCManager, SERVICE\_NAME, SERVICE\_START);

if (schService == NULL) {

// 打开服务失败

CloseServiceHandle(schSCManager);

return;

}

if (!StartService(schService, 0, NULL)) {

// 启动服务失败

}

// 成功启动服务

CloseServiceHandle(schService);

CloseServiceHandle(schSCManager);

}

# 三、服务的控制和管理

## 停止服务

可以使用ControlService函数发送停止请求到服务。

void StopMyService() {

SC\_HANDLE schSCManager = OpenSCManager(NULL, NULL, SC\_MANAGER\_CONNECT);

if (schSCManager == NULL) {

// 打开服务控制管理器失败

return;

}

SC\_HANDLE schService = OpenService(schSCManager, SERVICE\_NAME, SERVICE\_STOP);

if (schService == NULL) {

// 打开服务失败

CloseServiceHandle(schSCManager);

return;

}

SERVICE\_STATUS ssStatus;

if (!ControlService(schService, SERVICE\_CONTROL\_STOP, &ssStatus)) {

// 停止服务失败

}

// 成功停止服务

CloseServiceHandle(schService);

CloseServiceHandle(schSCManager);

}

## 删除服务

删除服务可以使用DeleteService函数。

void DeleteMyService() {

SC\_HANDLE schSCManager = OpenSCManager(NULL, NULL, SC\_MANAGER\_CONNECT);

if (schSCManager == NULL) {

// 打开服务控制管理器失败

return;

}

SC\_HANDLE schService = OpenService(schSCManager, SERVICE\_NAME, DELETE);

if (schService == NULL) {

// 打开服务失败

CloseServiceHandle(schSCManager);

return;

}

if (!DeleteService(schService)) {

// 删除服务失败

}

// 成功删除服务

CloseServiceHandle(schService);

CloseServiceHandle(schSCManager);

}

# 四、调试和部署服务

## 调试服务

调试服务通常需要在服务启动后附加调试器到服务进程。可以使用Visual Studio等IDE进行调试。

## 部署服务

部署服务通常包括将服务的可执行文件、配置文件等复制到目标系统，并通过安装程序或命令行工具安装服务。可以使用批处理脚本或PowerShell脚本自动化部署过程。

# 总结

通过以上步骤，我们可以使用C语言创建、安装、启动、停止和删除Windows服务。创建服务的关键步骤包括编写服务的主函数、实现服务控制处理程序、启动服务工作线程、注册服务以及服务的控制和管理。调试和部署服务也是确保服务稳定运行的重要环节。希望本文能对您创建Windows服务提供帮助。

**相关问答FAQs：**

## 1. 如何使用C语言编写Windows服务？

编写Windows服务可以使用C语言，你可以使用Windows API中的服务控制管理函数来创建、启动和管理服务。在C语言中，你可以使用Win32 API来调用这些函数，例如使用CreateService函数创建服务，使用StartService函数启动服务等。

## 2. 如何将C语言的程序设置为开机自启动的Windows服务？

要将C语言的程序设置为开机自启动的Windows服务，你需要创建一个服务，并将其配置为自启动。首先，你可以使用CreateService函数创建服务，然后使用ChangeServiceConfig函数来设置服务的启动类型为自动启动。这样，在每次Windows系统启动时，该服务就会自动启动。

## 3. 如何在C语言中处理Windows服务的开机自启动失败的情况？

在C语言中处理Windows服务的开机自启动失败的情况，你可以使用GetLastError函数获取最后发生的错误代码。根据错误代码，你可以采取相应的处理措施，例如输出错误信息、写入日志文件或者进行其他的错误处理操作。通过处理错误情况，你可以确保服务在开机时能够正常启动。