C语言如何调用打印机API

C语言调用打印机API的方法主要包括：使用Windows API打印函数、使用第三方库、直接发送数据到打印机端口、通过网络协议打印。其中，使用Windows API打印函数是最常见和推荐的方法。

使用Windows API打印函数是最常见的方式，因为它提供了丰富的功能和稳定的接口。具体来说，Windows API可以通过调用 `OpenPrinter` 、 `StartDocPrinter` 、 `StartPagePrinter` 、 `WritePrinter` 、 `EndPagePrinter` 和 `EndDocPrinter` 等函数来实现打印操作。

一、使用Windows API进行打印

使用Windows API进行打印是最常见的方法，因为它提供了丰富的功能和稳定的接口。以下是详细的步骤。

1. 初始化打印机

使用 `OpenPrinter` 函数打开打印机。这个函数需要传入打印机名称以及一个指向打印机句柄的指针。打印机名称可以在打印机和设备管理器中找到。

```
 #include <windows.h>

#include <stdio.h>

PRINTER_DEFAULTS pd;

pd.DesiredAccess = PRINTER_ALL_ACCESS;

HANDLE hPrinter;

if (!OpenPrinter("YourPrinterName", &hPrinter, &pd)) {

    printf("OpenPrinter failed with error: %dn", GetLastError());

    return -1;
}
```

2. 开始打印文档

使用 `StartDocPrinter` 函数开始一个打印文档。这个函数需要传入一个文档信息结构体。

```
 DOC_INFO_1 docInfo;

docInfo.pDocName = "Test Document";

docInfo.pOutputFile = NULL;

docInfo.pDatatype = "RAW";

DWORD jobId = StartDocPrinter(hPrinter, 1, (LPBYTE)&docInfo);

if (jobId == 0) {

    printf("StartDocPrinter failed with error: %dn", GetLastError());

    ClosePrinter(hPrinter);

    return -1;

}
```

3. 开始打印页面

使用 `StartPagePrinter` 函数开始一个打印页面。

```
 if (!StartPagePrinter(hPrinter)) {

     printf("StartPagePrinter failed with error: %dn", GetLastError());

     EndDocPrinter(hPrinter);

     ClosePrinter(hPrinter);

     return -1;

}
```

4. 写入数据到打印机

使用 `WritePrinter` 函数将数据写入到打印机。这个函数需要传入数据缓冲区以及数据长度。

```
 const char *data = "Hello, Printer!";

DWORD bytesWritten;

if (!WritePrinter(hPrinter, (LPVOID)data, strlen(data), &bytesWritten)) {

     printf("WritePrinter failed with error: %dn", GetLastError());

     EndPagePrinter(hPrinter);

     EndDocPrinter(hPrinter);

     ClosePrinter(hPrinter);

     return -1;
```

```
   }
```

5. 结束打印页面和文档

使用 `EndPagePrinter` 和 `EndDocPrinter` 函数结束当前页面和文档。

```
    if (!EndPagePrinter(hPrinter)) {

        printf("EndPagePrinter failed with error: %dn", GetLastError());

        EndDocPrinter(hPrinter);

        ClosePrinter(hPrinter);

        return -1;

   }

   if (!EndDocPrinter(hPrinter)) {

        printf("EndDocPrinter failed with error: %dn", GetLastError());

        ClosePrinter(hPrinter);

        return -1;

   }
```

6. 关闭打印机

最后，使用 `ClosePrinter` 函数关闭打印机。

```
ClosePrinter(hPrinter);
```

二、使用第三方库

除了Windows API，还可以使用一些第三方库来简化打印机操作。这些库封装了底层的API调用，使得打印操作更加简单和易用。

1. LibCups

LibCups是一个跨平台的打印库，支持多种操作系统。它提供了丰富的函数来操作打印机和打印任务。以下是一个使用LibCups进行打印的示例。

```
 #include <cups/cups.h>

int main() {

    http_t *http = httpConnectEncrypt(cupsServer(), ippPort(), HTTP_ENCRYPT_IF_REQUESTED);

    cups_dest_t *dests, *dest;

    int num_dests = cupsGetDests(&dests);

    dest = cupsGetDest("YourPrinterName", NULL, num_dests, dests);

    if (dest == NULL) {

        printf("Printer not found.n");

        return -1;
    }

    cups_option_t *options = NULL;

    int num_options = 0;

    num_options = cupsAddOption(CUPS_COPIES, "1", num_options, &options);

    FILE *fp = fopen("testfile.txt", "r");

    if (fp == NULL) {

        printf("Failed to open file.n");

        return -1;

    }

    int job_id = cupsCreateJob(http, dest, "Test Job", num_options, options);

    if (job_id == 0) {

        printf("Failed to create job.n");

        return -1;

    }

    cupsStartDocument(http, dest, job_id, "testfile.txt", CUPS_FORMAT_TEXT, 1);

    char buffer[8192];

    size_t bytes;
```

```
    while ((bytes = fread(buffer, 1, sizeof(buffer), fp)) > 0) {

        cupsWriteRequestData(http, buffer, bytes);

    }

    fclose(fp);

    cupsFinishDocument(http, dest, job_id);

    httpClose(http);

    cupsFreeDests(num_dests, dests);

    cupsFreeOptions(num_options, options);

    return 0;

}
```

2. WinSpool

WinSpool是另一个常见的库，专门用于Windows平台。它提供了对打印机和打印任务的全面控制。

```
 #include <windows.h>

#include <winspool.h>

void PrintFile(const char *printerName, const char *fileName) {
```

```
HANDLE hPrinter;

DOC_INFO_1 docInfo;

DWORD dwJob;

DWORD dwBytesWritten;

char buffer[4096];

FILE *fp;

if (!OpenPrinter(printerName, &hPrinter, NULL)) {

    printf("OpenPrinter failed with error: %dn", GetLastError());

    return;

}

docInfo.pDocName = "Test Document";

docInfo.pOutputFile = NULL;

docInfo.pDatatype = "RAW";

dwJob = StartDocPrinter(hPrinter, 1, (LPBYTE)&docInfo);

if (dwJob == 0) {

    printf("StartDocPrinter failed with error: %dn", GetLastError());

    ClosePrinter(hPrinter);
```

```
        return;

    }

    if (!StartPagePrinter(hPrinter)) {

        printf("StartPagePrinter failed with error: %dn", GetLastError());

        EndDocPrinter(hPrinter);

        ClosePrinter(hPrinter);

        return;

    }

    fp = fopen(fileName, "r");

    if (fp == NULL) {

        printf("Failed to open file.n");

        EndPagePrinter(hPrinter);

        EndDocPrinter(hPrinter);

        ClosePrinter(hPrinter);

        return;

    }

    while (!feof(fp)) {
        size_t bytesRead = fread(buffer, 1, sizeof(buffer), fp);

        if (bytesRead > 0) {

            if (!WritePrinter(hPrinter, buffer, bytesRead, &dwBytesWritten)) {

                printf("WritePrinter failed with error: %dn", GetLastError());

                break;

            }

        }

    }

    fclose(fp);

    EndPagePrinter(hPrinter);

    EndDocPrinter(hPrinter);

    ClosePrinter(hPrinter);

}

int main() {

    PrintFile("YourPrinterName", "testfile.txt");

    return 0;
```

```
}
```

三、直接发送数据到打印机端口

直接发送数据到打印机端口是一种较低级的操作方法，适用于需要精确控制打印内容的场景。

1. 使用Windows API

通过Windows API，可以直接发送数据到打印机端口。

```
 HANDLE hPrinterPort;

hPrinterPort = CreateFile("LPT1:", GENERIC_WRITE, 0, NULL, OPEN_EXISTING, 0, NULL);

if (hPrinterPort == INVALID_HANDLE_VALUE) {

    printf("CreateFile failed with error: %dn", GetLastError());

    return -1;

}

const char *data = "Hello, Printer!";

DWORD bytesWritten;

if (!WriteFile(hPrinterPort, data, strlen(data), &bytesWritten, NULL)) {

    printf("WriteFile failed with error: %dn", GetLastError());

    CloseHandle(hPrinterPort);
```

```
        return -1;

    }

    CloseHandle(hPrinterPort);
```

2. 使用Linux系统调用

在Linux系统中，可以通过系统调用直接发送数据到打印机端口。

```
     #include <fcntl.h>

    #include <unistd.h>

    #include <stdio.h>

    int main() {

        int fd = open("/dev/lp0", O_WRONLY);

        if (fd == -1) {

            printf("Failed to open printer port.n");

            return -1;

        }

        const char *data = "Hello, Printer!";

        ssize_t bytesWritten = write(fd, data, strlen(data));
        if (bytesWritten == -1) {

            printf("Failed to write to printer port.n");

            close(fd);

            return -1;

        }

        close(fd);

        return 0;

    }
```

四、通过网络协议打印

通过网络协议打印适用于网络打印机，可以使用Socket编程实现。

1. 使用Windows API

```
 #include <winsock2.h>

#include <ws2tcpip.h>

#include <stdio.h>

#pragma comment(lib, "Ws2_32.lib")

int main() {
```

```
WSADATA wsaData;

if (WSAStartup(MAKEWORD(2, 2), &wsaData) != 0) {

    printf("WSAStartup failed with error: %dn", WSAGetLastError());

    return -1;

}

SOCKET sock = socket(AF_INET, SOCK_STREAM, IPPROTO_TCP);

if (sock == INVALID_SOCKET) {

    printf("socket failed with error: %ldn", WSAGetLastError());

    WSACleanup();

    return -1;

}

struct sockaddr_in server;

server.sin_family = AF_INET;

server.sin_port = htons(9100); // Standard port for HP JetDirect

inet_pton(AF_INET, "192.168.1.100", &server.sin_addr);

if (connect(sock, (struct sockaddr *)&server, sizeof(server)) == SOCKET_ERROR) {

    printf("connect failed with error: %dn", WSAGetLastError());
```

```
        closesocket(sock);

        WSACleanup();

        return -1;

    }

    const char *data = "Hello, Network Printer!";

    int bytesSent = send(sock, data, strlen(data), 0);

    if (bytesSent == SOCKET_ERROR) {

        printf("send failed with error: %dn", WSAGetLastError());

    }

    closesocket(sock);

    WSACleanup();

    return 0;

}
```

2. 使用Linux系统调用

```
 #include <sys/socket.h>

#include <arpa/inet.h>
```

```c
#include <unistd.h>

#include <stdio.h>

int main() {

    int sock = socket(AF_INET, SOCK_STREAM, 0);

    if (sock == -1) {

        printf("Socket creation failed.n");

        return -1;

    }

    struct sockaddr_in server;

    server.sin_family = AF_INET;

    server.sin_port = htons(9100); // Standard port for HP JetDirect

    inet_pton(AF_INET, "192.168.1.100", &server.sin_addr);

    if (connect(sock, (struct sockaddr *)&server, sizeof(server)) == -1) {

        printf("Connection to printer failed.n");

        close(sock);

        return -1;

    }
    const char *data = "Hello, Network Printer!";

    ssize_t bytesSent = send(sock, data, strlen(data), 0);

    if (bytesSent == -1) {

        printf("Sending data to printer failed.n");

    }

    close(sock);

    return 0;

}
```

五、总结

C语言调用打印机API的方法主要包括：使用Windows API打印函数、使用第三方库、直接发送数据到打印机端口、通过网络协议打印。其中，使用Windows API打印函数是最常见和推荐的方法，因为它提供了丰富的功能和稳定的接口。此外，使用第三方库、直接发送数据到打印机端口和通过网络协议打印也是有效的选择，根据具体需求选择合适的方法可以更好地完成打印任务。