Window DK入门浅谈写给初学者 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/462/2021_2022_Window_DK_E5_c97_462573.htm 如果你是一个编程初学者，如果你刚刚结束C语言的课程。你可能会有点失望和怀疑：这就是C语言吗？靠它就能编出软件？无法想象Windows桌面上一个普通的窗口是怎样出现在眼前的。从C语言的上机作业到Windows编程确实有比较大的gap。或许你已经看了programming Windows的前三章，但是对于那个hellowin程序甚为迷惘。希望hfire的这篇文章能帮你填补这个gap，并提供一些学习的经验。为什么是浅谈，因为hfire知道的也不够深，肯定会有错误，请批评指正。 程序与运行时环境及操作系统 当你用Turbo C编写了一个C程序，然后编译连结它，得到了一个可执行文件。在Dos的命令提示符下键入这个exe文件的文件名，然后它就执行了。表面看事情就是这些。作为一个应用程序员，我们不用考虑背后的事情，但是有一个简单的事实我们必须清楚的认识：程序不只是靠自己运行，它需要运行时环境的配合。考虑一下用一个printf函数显示一个字符串的过程。显然这个函数不是你自己写的。或许你听说过C-Runtime Library，C运行时库，没错，你的程序只有依靠它才能运行。printf的代码就在C运行时库中，因此你可以轻松的调用它而不管它是怎么实现的。但是，C运行时库也会调用一些别的函数，这些函数是由操作系统提供的，称为中断服务程序，而操作系统的中断服务程序会进一步的调用BIOS中断服务程序。可以看出，程序的运行是由一层一层的服务支撑起来的。在这里面，操作系统担当了非常重要的

角色。它提供了程序员可以直接使用的例程，也可以称为Application Programming Interface (应用程序编程界面，API)。Dos中一般没有API的说法，Dos的编程界面是由中断服务程序充当。在Windows中编程就要常常和API打交道。32位Windows的API有2000多个，一方面它提供了功能强大的编程界面，另一方面它使初学者望而却步。 Windows操作系统基本常识 Windows是一个单用户多任务图形化操作系统。所谓单用户，指同时只能由一个用户（一个人）通过Windows系统操作电脑；所谓多任务，指同时可以有多个进程并发执行。既然Windows系统有这些特点，那么Windows编程就会体现这些特点。为了做到多任务，Windows程序使用消息机制，有我的消息我才干活，没我的消息就把CPU让给别人；为了做到图形化，Windows程序必须显示窗口并自己绘制客户区，就连显示字符串也必须画到客户区上。还有一点，Windows广泛使用动态链接。Windows的API就放在动态链接库中，以供程序运行时调用。在Windows 98中有32位的gdi32.dll,user32.dll,kernel32.dll和16位的gdi.exe,user.exe,krnl386.exe，API就存在于这些动态链接库中。 什么是Windows SDK SDK即software develope kit（软件开发工具包），它包含了进行Windows软件开发的文档和API函数的输入库、头文件（因为API在动态链接库中，这些动态链接库是系统的组成部分因此不用再提供，而输入库和头文件则必须，这样才能在你的程序中使用API函数）。早期SDK是一个单独发放的包，现在在Visual C 和其他一些开发环境中已经包含了它。如果你已经安装了VC 那么就可以开始编写Windows程序了。随着Windows系统的发展，SDK的内容越

来越多，我们只要抓住最基本的方面。至于其他专门的主题，就根据自己的兴趣和技术方向进一步学习了。 不用被第一个Windows程序吓住 如果你已经开始，你的教材应该是那本经典的programming windows (petzold)或者是一本相似的书。但无论哪本书，一开始你都会面对一个基本的Windows SDK程序，这个程序有几十行。尽管也不算长，但比C版的hello world长多了。更糟的是，里面充斥着奇怪的变量类型和常量定义，不过先不用被它吓住，让我们看看这里面有些什么。首先会有一个#include，嗯，没什么奇怪的，这和#inclde 没什么两样。然后是一个函数声明：LRESULT CALLBACK WndProc (HWND, UINT, WPARAM, LPARAM) . 有点困惑了，一下子就出来了好几个“生词”，而且函数名前面有两个修饰符也是以前没有遇到的。好在我们还可以辨认出这是一个函数声明。再往下看是WinMain函数，又是一堆生词，我猜想你可能已经开始郁闷了。想一遍看懂这个程序确实困难，所以看不懂也没关系。看不下去了可以看看书上的讲解。这篇文章并不是要完整分析这个程序的，hfire不可能有petzold讲的好。在这里hfire帮你分析一下一些陌生的东西。 首先说Windows的数据类型。尽管这些数据类型看上去很陌生，其实它们是由C的基本数据类型define的。比如UINT就是unsinged int,PSTR就是pointer to string 的意思,猜猜就知道是char * 。Windows还有很多系统定义的结构体，比如WNDCLASS,MSG等，这些东西见的多了就自然明白了。Windows还有一个重要的概念，句柄。通过句柄就可以操作Windows对象。HWND,HINSTANCE,HDC等都是句柄。 再说说Windows程序的结构。一般有一个WinMain函数作为程序

的入口点，在WinMain里面定义窗口类，进行消息循环。消息循环就是那个普通的while循环，在其中接收消息、分发消息。然后是窗口函数WndProc，名字可以自己定。在其中用一个大的switch结构检索消息，在每个case下面写处理消息的代码。最简单的Windows SDK程序只要写这两个函数就够了。等你的程序写长了，就要把特定的消息处理代码写成函数，以便在处理消息时调用，甚至你可以使用C 来写程序。等你熟悉这种结构以后，就可以任意发挥了。 其他的不想说太多，学SDK很重要的是不要期望在开始时把每行代码都搞清楚。 学习的方法 当然是多写程序了。最好每个主题都写一个。从一开始的窗口，文本显示到图形显示、控件、对话框，多写就能领会Windows编程的内涵。当第一部分学的差不多了，可以写一个综合点的程序。最后你会发现你可以写很长的程序了，1000多行也不算长，但对于当时学C时是难以想象的。还有重要的是多上一些专门的网站，比如VC知识库(www.vckbase.com)和VC之路(有一本很好的教程，不过现在网站的东西都没了，正在恢复)，另外www.csdn.net也是一个不错的地方。 几种Windows编程方法的辨析及其它 很多初学者往往会将Windows编程和VC 混为一谈。打开VC 的新建项目，可以看到VC 支持很多种工程。包括命令行的，MFC的，还有就是Win32 Application,即SDK程序。使用VC 未必是编写SDK程序，编写SDK程序也未必要用VC 。 编写Windows程序的方法也不止SDK一种，还包括使用类库如MFC,OWL,使用快速开发工具如VB,Delphi。这些方法各有各的用处。使用SDK无疑是最麻烦的了，我们学习SDK也并非是为了用它来编软件，虽然它可以，主要是为了对Windows编程有比较

清晰的认识，这样你使用MFC时就可以做到“ 胸中自有沟壑 ”。 