**VC及MFC概述和学习方法**

**一、VC是什么？学VC是学什么？**

**首先VC是一个软件（IDE集成开发环境）（编译、编辑、调试）**

**C和C++。**

**WindowsSDK（软件开发工具）**

**VC的灵魂：MFC（微软基础类库）**

**ATL（ActiveX模板类库）**

**其他的SDK，如OpenGL,DirectX,ActiveMoive,DrawDib(WinG)**

**VC是许许多多技术的综合，就好像少林72般绝技，一般在应用中学会其中很少一部分就够了。我们的现在目的，其实不过是学一套少林长拳。这套长拳应包括：**

**1、养成一种严谨的软件开发习惯，熟悉软件工程的基本原则。**

**2、C语言基础**

**3、C++最基本内容：类、继承性、封装性、多态性的概念**

**4、如何用向导建立一个程序框架**

**5、设计菜单**

**6、设计工具条**

**7、设计和使用对话框，熟悉最常用的对话框控件：按钮、静态文本、编辑框等**

**8、知道怎样新建类、成员函数、成员变量、消息处理函数。**

**9、了解最常用的Windows消息：如WM\_PAINT(重绘窗口)，鼠标按下、弹起、移动，初始化对话框，等。**

**10、了解最简单的GDI(图形设备接口)：写文字、画框、画点、画线、画圆，设置画笔、画刷。**

**11、熟悉最常用的MFC类：**

**CWnd**

**CDocument**

**CView**

**CDC**

**CDialog**

**CWinApp**

**CGdiObject及子类**

**CString、CPoint、CRect、CSize等简单数据类型**

**CFile**

**以上提到的这些内容，是每个人都会用到的内容。**

**二、MFC简介**

**上述主要内容都是与实际操作关系非常密切的，在此不可能讲清楚；而且，这些在一般的书里都会讲到。我主要将最重要的MFC介绍一下。**

**MFC是对WindowsAPI的封装，大大简化了我们的工作；学VC主要就是要学MFC，大约有100多个类，但常用的也就二三十个。应该象背4级单词一样将这些常用类搞懂；当然不要死记，要通过看帮助、看例子、动手练习来学会它们；而且，并非每个类的内部的所有函数都要学会，要日积月累。如果真的想成为高手，做个笔记本把自己认为重要的类、函数记下来，随时学习，也是很好的突击方法。**

**下面介绍最重要的MFC。**

**CWnd：窗口，它是大多数“看得见的东西”的父类（Windows里几乎所有看得见的东西都是一个窗口，大窗口里有许多小窗口），比如视图CView、框架窗口CFrameWnd、工具条CToolBar、对话框CDialog、按钮CButton，etc;一个例外是菜单（CMenu）不是从窗口派生的。该类很大，一开始也不必学，知道就行了。**

**CDocument文档，负责内存数据与磁盘的交互。最重要的是OnOpenDocument(读入),OnSaveDocument（写盘）,Serialize（读写）**

**CView视图，负责内存数据与用户的交互。包括数据的显示、用户操作的响应（如菜单的选取、鼠标的响应）。最重要的是OnDraw(重画窗口)，通常用CWnd::Invalidate()来启动它。另外，它通过消息映射表处理菜单、工具条、快捷键和其他用户消息。你自己的许多功能都要加在里面，你打交道最多的就是它。**

**CDC设备文本。无论是显示器还是打印机，都是画图给用户看。这图就抽象为CDC。CDC与其他GDI（图形设备接口）一起，完成文字和图形、图像的显示工作。把CDC想象成一张纸，每个窗口都有一个CDC相联系，负责画窗口。CDC有个常用子类CClientDC（窗口客户区），画图通常通过CClientDC完成。**

**CDialog对话框**

**CWinApp应用程序类。似于C中的main函数，是程序执行的入口和管理者，负责程序建立、消灭，主窗口和文档模板的建立。最常用函数InitInstance（）：初始化。**

**CGdiObject及子类,用于向设备文本画图。它们都需要在使用前选进DC。**

**CPen笔，画线**

**CBrush刷子，填充**

**CFont字体，控制文字输出的字体**

**CBitmap位图**

**CPalette调色板**

**CRgn区域，指定一块区域可以用于做特殊处理。**

**CFile文件。最重要的不外是Open（打开）,Read（读入）,Write（写）**

**CString字符串。封装了C中的字符数组，非常实用。**

**CPoint点，就是（x,y）对**

**CRect矩形，就是（left,top,right,bottom）**

**CSize大小，就是（cx,cy）对（宽、高）**

**三、用好MSDN和例子**

**作为提高，推荐看的例子有：**

**nHello最简单的程序框架**

**nScribble鼠标绘图。教程：编程基础，GDI**

**nDiblook图像处理最简单范例**

**nMFCIE因特网浏览器**

**nEnroll数据库**

**nDaoviewDao(数据访问对象）范例。树型和列视控件**

**nCubeOpenGL范例，画一个不断旋转的方块**

**nHierSvrOLE服务器**

**nOclientOLE包容器**

**nDrawcli鼠标绘图。也包括OLE。**

**nWordPad写字板。OLE**

**四、几点经验**

**A.为什么要用Windows而不是DOS编程**

**Windows编程并不比DOS编程难很多，只是在刚刚接触的时候感到头绪繁多，不知从何下手。实际上，如果你会C语言，你会发现Windows编程的难度并不比当年学C语言时大。DOS编程要想成为高手，也需要学习几百个函数（无论是TC、BC、MicrosoftC），并要深入了解汇编语言级的DOS功能调用。而现在在Windows下，付出相同的工作量，你却可以获得几倍的收益。如果你以前熟悉DOS编程，现在再搞Windows编程，只要两个星期肯定会感到巨大的优越性：在DOS下，每一件事都要自己干，而且还有64K内存、显示精度等的限制，实在是事倍功半了。**

**B.为什么要用VC而不是其他Windows编程语言**

**n大多数大型软件（包括Windows自己）都是用C、C++编的，所以可以利用的源代码特别多。例如Photoshop，就有VC的编程接口。的其他的语言都不如VC有历史优势。而且VC的开发工具特别多，从控件公司到硬件开发商，其产品很少敢不提供VC接口的。**

**nVC适合组织大工程（VB就不行）**

**nBorlandC++或C++Builder也很不错，并不比VC差。但BC的版本兼容性不好，从3.1到4.0就出现“代沟”，而且VC提供了极为丰富的文档和范例,VC的参考资料也远比BC多。并且VC与其他的VisualStudio成员可以协调工作，对软件工程有利。**

**C.开始不要学WinAPI编程，但以后最好学学WinAPI**

**特别要注意，开始不要买讲WindowsSDK编程的书。一个臭名昭著的例子就是“HelloWorld”。各种计算机语言在入门课中往往都给出一个例子，显示一行文字“HelloWorld”，无论是在Basic、C、Pascal，甚至汇编中，这个最简单的小程序都几行就可以写完。但在早期的Windows编程中，刚才讲到的“HelloWin”这个有名的例子就要组织一个工程、写上几百行代码才能实现。这就吓跑了很多想学Windows编程的人。实际上，在Windows下显示“Helloworld”,最简单的方法只要5行就够了：**

**#include“windows.h”**

**voidWinMain()**

**{**

**MessageBox(NULL,”HelloWorld!”,”MyFirstWindowsApplication”,MB\_OK);**

**}**

**当然这个程序只建立了一个最简单的对话框，但这已经够了。**

**如果你没有SDK编程的经验，学VC就是要利用MFC绕过Windows编程的许多细节。有好的工具为什么不用呢？现在你在VC写一个”HelloWorld”，只要写一行代码就可以了。**

**voidCYourView::OnDraw(CDC\*pDC)**

**{**

**//…….其他的代码**

**pDC->TextOut(100,100,”HelloWorld!”);**

**}**

**但MFC不是万能的，WinAPI和VC中有很多功能在MFC中并不能反映出来。在入门后，要想发挥VC的完全能力，WindowsAPI还是非常重要的。而且，这些API在其他Windows语言中也用得到。**

**学语言，与练武道理也差不多。VC、VB、Dlphi、VFP，以及其他的Windows编程语言，都属于同一个门派：Windows门，虽然招式不同，但内功都是一样的，这就是WindowsAPI(应用编程接口)，大概是两千个函数。任何武功，招式即使再复杂，通常也能在很短时间内记下来；但内功就不是一朝一夕的事。而且，各门派的内功不同，Unix、Linux门有Uinx的内功，Apple(苹果)门有Mac的内功，Java更是自成一派。我们这里说的学VC，只是学招式，学招式一个月差不多就够了，聪明的两个星期就可以掌握。光学招式，吓唬人是够了，但和高手一过招，非出丑不可。要想精通VC，必须熟悉WindowsAPI。**

**D.高级语言与自然语言**

**一门高级语言与一门自然语言是类似的。关键要素有三：**

**语法：了解C和C++基本语法，这是比较容易的。**

**修辞：编程的艺术和风格。通常，这是与软件工程联系起来的。算法上的小伎俩并非是关键的。严谨，严格按照规范去做，同时又能发挥出极大的创造性，才是语言学家的本色。**

**词汇：如何利用别人的工作？如何利用自己以前的工作？这是一个软件工程师成功的关键。如果词汇量不丰富，就写不出好文章，就会有想法表达不出。一个人的语法在上小学之前就基本上学成了，修辞水平也在年轻时就确定了，惟有词汇量需要根据需要不断学习、不断扩充。VC编程中，拥有大的词汇量就意味着要掌握较多的类、函数和其他编程组件。如：WindowsSDK,MFC,以及你所专攻的领域的现成的SDK，比如3D编程中的OpenGL,多媒体编程中的DirectX，图像处理中的Victor库。另外，ActiveX控件的掌握往往也是成败的关键。我们这几年的工作也总结成了一个类库，叫IPX（图像处理框架），大大简化了图像处理的工作；不过现在还有很多细节问题要解决，如果有同学感兴趣可以参与这方面的工作。**

**学习VC，千万不要用我们在学校里学英语的方法，而要用小孩学说话的方法。所谓学英语的方法，就是我们在本科时学C语言的那种模式：详细搞懂每一个细节，背大量的规则，很少上机实践，没有实际问题供解决，最后由考试来断定学习成果。这种模式的后果大家都清楚：我们学了十几年英语，可有不少人还只会四五千个单词，写百把字的文章，不会说，不能听；尤其要命的是，看不懂VC的帮助文档。我们要象婴儿一样，在使用中学会语法、修辞和词汇；一开始肯定只能编出最糟糕的，就好像小孩开始只会说：“爸爸坐车车”；但“爸爸”“车车”这些词对小孩是最需要的，学习的效率反而最高。学VC的时候，应从我的需要出发，先做一个最简单的东西，然后一点一点复杂起来，象滚雪球一样。**

**学习VC尤其要和别人交流，你苦苦思考不明白的，也许别人早就解决了。一定要善于学习：从书店找出有用的书；向身边的高手学习；寻找可以利用的源代码、类库、组件、控件、库；如果能上互连网，那就更加能解决问题了。**

**如何学好VC和MFC（各前辈学习方法及感受整理）（一）**

**序：**

近来,在论坛上面几乎每天都可以看到很多的人提出如何学习MFC,学习MFC的确不是一件很容易的事情,加上成千上万的API函数和微软的一些资料的保留,学习MFC就变得更难了。不过我认为难在入门，入门后就慢慢觉得得心应手了。下面是一些先辈们学习过程中总结出来的方法及学习感受，今天把它们整理出来，供大家参考。

**正文：**

我记得我在网上是这么说的：先学win32的SDK,也就是API, 再学MFC，这么一来呢，就先有个基础，MFC是API的封装， 如果API用的熟了，MFC上手就轻松的多了。

我觉得呢，现在不少WINDOWS编程入门的好书很多，多买一些，多看看，多想想。不要心浮气躁。我的经验就是一边看书，一边写程序，看看是不是书上说的那样。看书看明白了不行，自己做对了这个过程体会就不一样了，做的多了，就慢慢会有“悟”的感觉。就入了门了。一般从刚开始学WINDOWS编程（前提是C/C++至少要懂得C的一般常用概念）到基本熟悉，一年时间也差不多，如果在公司里，可能会快一些，主要取决于自己，如果每天都很多时间看书就要快许多。

这是昨天一个网友提出和你同样的问题。我这样说的。你说推荐的书籍呢，那倒真是有不少好书非常值得看看，学习c++的有《c++编程思想》，学习api就是win32程序员参考手册，一般叫sdk。学mfc呢，那就更多了，书店里的不少书都可以拿来看看，看的多了，自己也就知道什么样的书最适合自己，不同时期欣赏的口味都不同的。比如说刚开始，就特别想找vc如何使用的书，刚接触win32编程就想如果能有手把手那样的教就好了。这些书都有的。我的习惯就是，多看书，然后才有鉴别，知道什么样的书算是经典，什么样的书尽管没多大用，但是对特定的人，特定的时期，特别有用。   
买更多的书，看更多的书   
其中《Visual c++ 技术内幕》和《深入浅出MFC》不可不买   
中国人学不好VC的原因很简单:英语非常差,不会看英文版的MSDN/CD,大学里的教育全是垃圾，设备又差，许多学VC的家伙不懂MASM,连WINDOWS Message cycle system 都不了解,学什么学???不过不要紧!据我所知,全球使用C++的人不过30万,且多数是用BCB(用控件来写容易啊!!!???)要知道，根据MS的资料,它出的书最High class 的一类只不过卖出20万左右，(ps:I forgot bookname!!!)要想学好VC，先学好英文，再去   
Codeguru.com之类的网站D/L source code !不要买国人自己编写的书(翻译的除外)最好是MS出的书，既然是用MS的工具,这是当然的,还要定期买MSDN/CD,MS的人开发人员每人一套!!!!要谈的太多了,一句话:   
学好VC=英文好+基本功好+勤奋+聪明.少一不成的啊!!!!   
我觉得初学者先不要看MFC,API，Windows编程之类的东西，最重要的是先弄懂什么是面向对象，什么是类，要是弄不懂类拿学起来就真的累了！   
1.先找一些傻瓜书，一步一步教你，CLASSWIZED,单击，双击，右键...,...加入函数代码....编译,运行...(真正的傻瓜!)   
2,分析代码,头文件,消息宏等,找来书慢慢研究吧.哦,这函数有这些参量.....哇,老兄,WINMAIN()是做什么的???????指针,类,继承,重栽,折构......   
3.书书....<>,95元,<<深入C++类库编程...>>,SAME图书,什么,105元... 清华,希望,机械,电子,什么出版社......

说起来我学VC也有些年头了，想当年玩了3年游戏却在大四为了毕业设计开始接触Windows编程。没办法，天生的手短心长，导师原来要我用DOS环境的。偏偏一开始有抱的是MFC，有一点C++基础的我看得是两眼发黑。当然，设计是泡汤了，还好导师放我一马。   
工作以后才切身感受到，学Windows编程是要从API，从SDK玩起的。这还是我当年去一家台湾游戏公司面试的惨痛教训。我那个时候因为默不出一个完整的WinMain()和WindowProc(...) Windows SDK程序架构惨遭淘汰（其实现在大多的Windows游戏还是用SDK的）。   
在我看来，写Windows程序的人对API不了解是不可想象的。我现在还没有用VC编出一个我自己认为象样的东西来，但我还是坚持用我的业余时间去和她增进感情，也许，这是一个技术人的老毛病吧。刨根问底，对Windows也不放过。   
还有，就我觉得，语言这个东西其实就象老妈厨房里的各种菜刀，各有千秋的。人天生的对东西会产生偏好，但拿水果刀切菜和拿大菜刀削水果是一样吃力的。我以前也认为天下有了VC就已足够，可现在两样。   
有人说过，存在的就是合理的......   
各位友人,我觉得学VC,熟悉C++的面向对象与WINDOWS编程技术是必要的.另外多看看别人的源代码,弄清程式所来龙去脉也,弄清程式的运行机制格外重要.至于你不太熟悉API,我看这并不重要,因为这么多API,试问谁能记这么多,不知道的查MSDN.初学者切莫心浮气燥,循序渐进,相信他日   
你也是VC"高手".   
不过,想精通VC还没有这么简单,我以什么VC,VB,BCB...   
只不过编码工具,大家不必拘于此.VC确实是一把好剑,像独   
孤九剑也是一把剑,但没有一点内功,这把好剑不是那么好用的.   
我觉得学任何语言都无所谓，关键是要掌握程序设计的核心——算法和数据结构。各种语言和开发工具就好比各门各派的武功招式，而算法和数据结构则好比内功心法和武学原理，内功高强了，任何招式到了手上都能化腐朽为神奇，掌握了武学原理，则能无招胜有招。所以，建议大家多花点时间学习基本功，一味地追求各种开发工具的开发技巧是没有意义的。因为各种开发工具会不断地更新，比如今天你在学C++，明天微软搞了个C#，你又得跟着学C#。但是如果基本功扎实，掌握了程序设计的核心思想，不论什么语言都可以很快上手，无论何种开发工具学起来都事倍功半。我并不是否认学习各种语言和开发工具的必要性，只是要提醒各位初学者，学会C++语言或学会用VC编程并不等于学会了编程！编程的关键还在于数学模型的建立，数据结构的选择和算法的设计！   
内功=算法+数据结构+编译原理+\*作系统原理+软件工程+英文   
你到 [http://www.csdn.net](http://www.csdn.net/) /;;看看，可以找到《深入浅出MFC》2e电子版下载！   
有一本书很好《WIN32程序员参考大全》即可了解WIN C SDK，又可以了解WINDOWS，一举两得。就是有好几本。可以到VC知识库网站的电子图书栏目下载。   
我认为需要三本书，   
1 <> 美 Jeffrey Richter.清华大学出版 99元   
2 <> 美 Charles Petzold 北京大学出版 160元   
3 <> 美 Jeff Prosise 是英文版的 北京大学出版 198元   
还有更好的选择吗？我不相信！

我学vc也半年了，c和c++的功底不错，对vc感觉还可 以。我认为学vc也有些技巧，可以事半功倍。 vc的书要跳着看，先把api,windows运行的概念看懂， 再看mfc其实只要你c++学的好，mfc几乎没多少可看的， 就是一些功能的封装。vc其试只是工具，精华都在windows   
程序的流程上。 掌握好概念后，就是狂背各种api函数和mfc封装的 成员函数。 最后就是研究mfc代码和其它原代码。

我学vc已有些时间了。但总有的地方不清楚。到网上去问“高手”，老被骂的狗血碰头，到头来，还是一头雾水。后来买到一本好书，   
The MFC Answer Book, Solutions for Effective Visual C++ Applications   
Author: Eugene Kain   
Published by Addision-Wesley   
个人认为学习VC的前提是C＋＋，特别是虚函数、派生、和指针，余皆不足虑也。 vc 的瓶颈就在于各种api和mfc封装的函数调用。例如你知到 ddx在update()运行时默认凋用dodatachange(),等等一些函数刁用。vc顺脆是小菜。 vc本身就是傻瓜工具，微软开发vc原意就是要缩短 程序员培循时间。可国内有人据燃要用几年时间来学，真是太可怕了，太可悲了。学会了只能在windows平台上开发，那末你那有时间去学其他平台的程序开发。   
vc只是工具，关件在学windows\*作系同，windows 通了，vc最多半年。 劝大家一句，不要在看身末“深入浅出”之内的 书了，去找一本专门讲vc库函数的书看，这才是精通之道。   
住：以上经验只试用于，具有很强c/c++遍程能力者。

我觉得学vc要先从c，c++学起，然后是api和mfc(英语很重要，不过不用怕，用金山词霸边查边学，等你vc入门后，你的英语阅读能力也会上一个台阶);   
另外我要的经验是学的时候不要钻牛角尖，学语言关键是   
学语法和概念(继承，重载，多态性...)，其次才是某某函数怎么用之类的东西.   
(学windows编程一定要掌握消息机制)   
要想学懂VC必须：   
1、找几本好书：《window程序设计》只看前几章，看懂   
windows的消息机制，Api的结构；   
《mfc深入浅出》全面学习，深刻理解mfc的结构   
2、编一个较大型的项目，不停的使用VC，边用边理解看   
过的东西；   
3、经常看优秀的源程序。   
我本来无资格发言的，因为我也是一个多月前才决定学VC的。我总结一下以上n位的发言(随便练练指法)：   
1.在学vc之前最好能有点其他语言的基础.(没有也没多大关系)   
2.学c++之前不必学c.(当然会c会快点)   
3.c++的一些概念很重要(如:对象,类,构造、析构函数,继承,模板等),最好弄透(怎么弄透,谁告诉我?)好书推荐:<>(<>screen.width-300)this.width=screen.width-300'>39.00元,适合对c++较熟悉的人.(有没有适合白痴级的好书,让我弄透以上重要概念,麻烦介绍一下)   
4.API---->MFC.好书推荐:   
(1)<<深入浅出MFC>>(<>screen.width-300)this.width=screen.width-300'>80.00元,701页.电子书下载:[www.csdn.net](http://www.csdn.net/)(不过是繁体版的,有些术语和大陆不同).购买:[www.huachu.com.cn](http://www.huachu.com.cn/). [www.beijingbook.com](http://www.beijingbook.com/), 人民时空.(第一家最便宜)   
(2)<>(<>screen.width-300)this.width=screen.width-300'>96.00元,900多页.最好买第四版的(清华出版,潘爱民翻译),电子书下载:至今也没找到(只有第五版,English的),一些网站上说的好听其实都是些原代码(害我下了100多兆的原代码,这些人建网站干吗,吃饱了撑地,这里copy一些,那里copy一些不看一下就上传,有什么意思) .购买同上.   
(3)<>(<>screen.width-300)this.width=screen.width-300'>(第五版,以前的版本叫<>我没找到电子书([www.ssreader.com](http://www.ssreader.com/)好像有)和购买网站)160.00元,1346页 电子书下载:至今也没找到(只有第五版,English的), .购买同上.   
以上的书有人看懂了,说好   
有人没看懂,烦恼   
fang说:什么样的书尽管没多大用，但是对特定的人，特定的时期，特别有用   
我说:尽信书不如无书   
5.边学边练,多看MSDN,英语不好的人,会费尽一些(我算是完了,真的很傻,以前马老师对我不错,想起来真对不起他(对不起的是你自己))   
学习VC，MSDN既重要也不重要，关键是先后问题。建议你先看 侯俊杰的《深入浅出MFC》，买不到的话可以到我的主页的电子书下载理去下载。要是你是成都的话就比较幸运些，成都有这本书卖哟！！！相信看过这本书后，你就知道该怎么学VC了。MSDN在电脑城去买，4元一张。一共3张（新版，老版2张）。   
好了，先把上面做了！快行动哟！！！   
我是一个大一的学生，读计算机的。我对编程很感兴趣，我以为可以在学校学到好东西，于是就慢慢来学，现在差不多有一年了，谁知连个屁也没学到。直到我来到这，被大家的热情感动了。我决定自学。后来我听说有些大学生能通过高程。于是我就买了[离散数学],[C程序设计],[计算机英语]学起来.听说还要学[数据结构],[\*作系统],[编译原理].请问要学完这些要多少时间?先声明我是一个菜鸟.还有应该先学哪些?我正在为此奋斗,希望大家指点小弟一下.   
学VC我认为有益下要点; 1.要对win95\*作系统的特点了解。如：多任务，消息映射等。 2.面向对象的设计特点，即类,需要学好C++。 3.对MFC也要有一定的认识。 4.真对一实际问题深入编程，不投机取巧，老实下功。   
过去我在VC上下过很大的功夫，光MFC,WIN32API手册就看过很多遍。我认为学好VC的关键在于对Windows 系统的了解，如果连某个动作对应的消息都不知道，又如何处理它了。

**如何学好VC和MFC（各前辈学习方法及感受整理）（二）**

VC 进阶：1. C 语言；2. C++ ，OOP；3. Windows 运作机理，SDK编程；4. MFC类库，应用程序框架编程；5. ActiveX，ATL，COM……每步骤通常需要6个月，最快也得3个月，诸位同意否？   
学 VC 或 windows 编程分三 STEP:1. 只 SDK 提供的 Function 作开发，不要Object, 你会知 How to handle Message. 2. MFC 编程, 用 CEdit 等开发.3. 再用 VC如果你有一个别人的好程序， 不妨自己跟着他的思路，从小到大自己完成一遍。好书：<> 学习VC++有相当的内容要学，而最后的深度则看个人的悟性与勤奋了。1）掌握最新标准的C++。 如果曾经在大学里自以为学过C/C++,还对dynamic\_cast/static\_cast/template/try/catch/throw/stl/...感到陌生，那你该Refresh一下新的ANSI C++标准了。 ----此与VC++无甚关联，g++/bcb均有所支持。（2个月)2) 学习SDK编程；:-O （6个月） 痛苦是暂时的，必要的，坚持就是胜利！ 建议用Lccwin32/Masm32/Tasm编写小而精的工具软件； "Windows Programming Guide." "Advanced Windows programming "3) 研究MFC源代码。(6个月) 不要被一大堆的宏所蒙蔽，just track and dig into it!! 在知道MFC如何扩展，包装SDK之后，自可功力渐进，不被其MFC系统框架所困绕。 "MFC Internals"4) 研究OLE/COM技术。 COM/ActiveX技术是MS的核心技术，只有彻底洞察其理论精髓才可以体会现在的\*作系统的技术趋势，以不变应万变。 "Inside Ole 2"\*\*\*几点建议\*\*\*：1) Delphi/BCB/VB可以很快构筑界面，但对于想真正学习、理解系统不是一个好的平台，但如果有时间读一读VCL的源码，看看Borland是如何封装系统的，也可借鉴不少。2) 学习ASM对理解C++有很大益处。Soft-ICE也是和VC++结合在一起的好工具;3) 语言本身是皮毛，算法是筋骨；无论高级语言如何发展，在理解\*作系统的基础上不断自我创新的能力是源源不断的；如果精髓一旦了然于胸，选择何种开发工具都可以驾御，一见如故了。

VC入门的一条路   
首先声明，我可不是什么高手，也就是刚脱贫。因为运气很好，买到了几本好书，在学习过程中几乎没走什么弯路，没费什么劲就入门了，现在一般的小程序能搞定了。看到那些因为不知从何下手而苦苦挣扎的朋友，希望我的文章能给他们一些帮助。   
  
学编程急不得，上来就学VC肯定碰一头灰，说VC难就难在这点上了。如果硬上，意志坚强的话也许能挺过来，但也是会缺乏后劲，不得不回过头来补习基础知识。意志不坚强的话，很有可能就此放弃了，并留下一个VC难得不得了的印象。其实，只要踏踏实实一步一步来，VC也就是很简单点事。在这里我说一下，如果你还只是一个初中生，那么你就应当仔细考虑一下了。首先，限于你的知识和思维能力，学学C语言还应该没问题，但要学VC是要下相当大的功夫的。而且，你现在学到的东西将来一定会过时，所以不如把精力放在算法的研究上，毕竟这些东西永远都不会过时。   
  
我觉着如果走对路子，入门VC，一个暑假应该够用了。   
  
虽说学VC并不是传说的那么难，可不下些苦功夫是学不成的。在学VC前，你必须确定，你是因为热爱编程才学VC的，而不是出于炫耀或其他。否则，恐怕你坚持不到胜利的那一天。   
  
对于首次接触编程的同学，我建议最好先学一下Quick BASIC（DOS里就有带，就是那个QBASIC.EXE）,不用多学，知道什么是整型什么是浮点，以及DO...LOOP FOR...NEXT 等最基本的知识，能算个阶乘，求个积分就行了，关键要明白计算机是怎么执行命令的，对编程有个感性认识。如果你对自己有自信，也可以直接学C++,应该也不会多费太多事。不过我觉着因为有很多资料是FOR VB的，并且网页编程常用VB Script,以后你也很有可能要用到VB，所以了解一些BASIC对以后也是有帮助的。而且如果只是要算个小题用QB比VC、VB之类方便多了。（找个人教一两天应该就能搞定了吧）   
  
之后我建议找本初高中信息学奥赛的书（有些奇怪吧）。这些书里讲解了基本的算法，写的却比较通俗，没什么高深的数学知识，容易理解。这些东西在数据结构课程中有系统严密的讲解，所以也可等以后再学。但我觉着现阶段写一些小程序，实现些常用算法对自己的思维是很好的锻炼，对以后的学习大有好处。我想，能排排序，求个八皇后，算个汉诺塔，再知道深、广度优先搜索就可以了。   
  
如果你已经有一些编程的基础知识了，自然就可以跳过以上两步。   
  
等你对编程有一定的认识后，就应该开始进军C++了.注意,是C++而不是VC,这两者根本不是一码事!别买VC入门,你肯定看不懂！C++是一门语言，而VC教程则是讲解如何使用MFC类库，学习VC应建立在充分了解C++的基础上。   
  
我觉着比起许多花花绿绿的大全、宝典，不如买本大学的C++教程（注意：别买等级考试的教程，那是应试用的），那些教程大都经过反复锤炼，比较系统，很有含金量，有不会太贵。这里我暴力推荐由清华大学出版社出版的一本《C++程序设计教程》，封面是红色的，封底是黄色的，看起来很朴素，主编是钱能。这本书简直是好的没法说，还便宜（39.5元）。这本书不但告诉你语法，还告诉你为什么要这样规定语法，看完之后，你会觉着那许许多多的规定是理所当然的，根本不用费心记,不但不枯燥，反而会使有一种美感。书中还有许多实际编程时应注意的问题，很有价值。这本书还配有《实验指导》，和《习题集》，不过对于有一定编程经验的人应该是用处不大，所以我没买。除了这本书，我建议再买本关于传统C语言的书，要着重以下几个方面：编译预处理、typedef的使用，位运算，标准库函数的使用（最好有一个比较详细的列表，用的时候好查）。   
  
C++的标准类模板是个不错的东东，不过对于学习MFC却没有帮助，原因是MFC太落后了(^\_^),可以先放一放，要不要学等搞定MFC时再作决定。虽然我没怎么学过，不过觉着它挺不错，而且代表了一种发展方向。虽然学着费劲（那东西几乎没一点C++的样子,简直是在学新语言），但掌握它可使编程轻松不少。   
  
学完了C++,该学VC了吧？别急，先找本WIN32编程的书，这次没什么讲究，看着差不多的,便宜一些的就行了，如果有对于VC集成环境的介绍就最好了。WIN32编程是指用WINDOWS API编程，MFC是将众多的API函数进行了封装的类库，用起来方便了不少，但如果没有一些API编程基础，对相关概念不了解，学起MFC来可能会很费解。相信我，将来你不会觉着学API编程是浪费时间的。API函数有两千多，要都学会累死。其实你只要知道如何创建窗口、对话框、控件、消息循环，了解了句柄的含义就差不多了，最好再了解一点GDI编程。这个阶段的关键是要理解WIN32程序是如何运作的，把握各种窗口间的关系。因为你以后应该很少直接用WIN32 SDK编写程序，所以关键是理解整体结构，记住几个函数并没有什么意义。函数用的时候可以现查。手头应有一份比较全面的资料。中文的API函数大全好像只有For VB的，不过看起来应该是没有任何障碍。英文的资料MSDN中就有，相当全面，也比中文的详细一些。我建议用中文资料快速查找，有问题再看英文。（我忘了我的html版中文API大全是从哪下载的了，不过书店有售，白皮的，很厚一本，很贵就是了。谁想要可以与我联系iamsor@sohu.com）

对于想学游戏编程的朋友注意：从现在起，你不必去学什么MFC了，相信我，那东西除了让你头疼一点用处都没有。游戏都使用Win32 SDK编出来的。现在，你要学的是DirectX。我推荐老王翻译的DirectDraw文档（[www.imagic3d.com](http://www.imagic3d.com/)），前面有一些老王自己写的基础知识，很适合入门， 在此特别感谢老王的辛勤劳动。我还买了一本《DirectX 7 速成教程》，机械工业出版社出版，封面封底都是红色。除了翻译惨了点还算不错，介绍的挺全面，还有一些游戏编程基础知识。如果你想图省事，可以考虑用云风编写的风魂游戏库，很方便的(www.codingnow.com 对云风的辛勤劳动表示感谢).不过我仍然建议至少了解一下DirectX.这里说一下，很多朋友抱怨找不到有关DirectX的书籍，我告诉你，别去编程书籍里找，去多媒体制作书籍那找，还要把眼睛瞪得大大的、圆圆的。Director.. Director.. DirectX!!!知道了吧，书店工作的人眼神都不大好，我也是有一天没事瞎转才发现的，吓了我一大跳。发现这个秘密后我发现其实有不少关于DirectX的书，只是一般人找不到罢了。学会了DirectX，就可以开始奋斗了，剩下的问题就只有积累经验了。   
  
对于志向是应用软件的朋友，打好了基础，现在进军VC的障碍都已扫清了。不过还是应做好充分的思想准备，事情不会那么顺利的。刚开始，你就要面对一个瓶颈，大量的紧密关联的知识，使你必须一下理解一个整体，而很难一点点来。可以说，学VC最难的就是刚开始。情况有些类似于你第一次创建窗口时面对的那长长的程序,只不过更糟糕就是了。很多人就是在此被一大团知识噎着，放弃了学习。事实上，这几乎是Windows编程的一大特点，在WIN32编程时你就应当有所体会，而DirectX和COM都有类似问题。记住，一定要坚信：突破了这个瓶颈之后便是一片海阔天空，而有着扎实基础的你是一定能闯过去的。   
我的建议是，不要等彻底弄明白一句才看下一句，而是知道好像怎么回事就向下看，实在看不懂也硬着头皮往下看，等能比猫画虎的作出个菜单、对话框时便回过头再看一遍，几遍之后应该就可以大概明白MFC程序的运作了。注意：这里有个关键，一定要用心体会面向对象编程的精神，努力将眼前的程序与你之前所学的知识联系起来。通过比较成员函数和对应的API函数间的区别，尝试去理解MFC是怎样封装相关的API的。例如，相当多的类将句柄封装了起来，在传递参数时加以省略，这就是一个很重要的特性。MFC中有许多的宏，看起来很奇怪，但你不必理会其实现方法，只要知道如何使用便可，这点也很重要。还有，千万不要过于依赖ClassWizard，对于其生成的代码要自己进行分析，偷懒就不能真正理解MFC。   
  
这里我介绍一种学法，不过学法这种东西因人而异，所以还要根据个人的情况决定。开始时，教程让怎么办就怎么办，比猫画虎的作出个菜单、对话框，然后试着添加控件。等能不看教程便做到这些时，开始逐字逐句的试着理解MFC程序的组织。若看不懂就跳过看下面的。就这样一遍一遍来。如果觉着头大就先歇上两天,等清醒了再接着来.最好在这期间多找些介绍MFC框架的文章。注意：先不要碰有关文档／视结构的东西，否则你会更头大。然后便可试着写个定时器、名片夹之类的小东西，熟悉一下各种控件的使用。   
  
突破了开始的瓶颈之后再学下去便会越学越容易，因为你可以一点一点的积累知识，而不用担心被噎着了。再过不久，你就不得不开始使用API函数完成一些功能，如在系统托盘加图标，自画列表框等。这时，你也会体会到学习WIN32编程的好处了，没学过的人很可能就此卡壳。   
  
关于VC的教程遍地都是，但鱼龙混杂，市场乌烟瘴气，骗钱的书真不少。有些书很厚，内容当然也很全，对老鸟很不错，但对于初学者似乎有些浪费钱，因为里面绝大部分的东西MSDN中都有，并且初学者用不着。老外写的书一般内容不错，但翻译水平是个问题。如果翻译糟糕可够你受的，加上价格不菲，我也不建议初学者买。还有些什么傻瓜书、图解之类，好像是面向初学者，时则为骗钱，你看完后除了那几个例子什么也不会作。如果你看到有一本书花了大量的篇幅图文并茂，生动形象的介绍如何画图，输出文字，别买！骗钱的！（:-< 我就被坑去42块现大洋）那些花哨的东西对初学者除了转移注意力没任何好处。如果你真的对GDI绘图很感兴趣，就去找些专门书籍，比那些书好多了。最好买名字类似于VC应用，VC编程之类比较正式的，图不要太多的，字不要太大的，最好是有较多的控件介绍的。当然，文字流畅也是很重要的。书不要买太多，最多两本，因为这些书内容都差不多，买多了的话东看西看反倒没好处。在这里，我建议初学者先别急着掏钱，而是去VC王朝（ <http://vcdynasty.yeah.net/>）下载本MFC教程。虽说是网上教程，却当相当有水准。不但内容比较全面，还一句废话也没有（这里感谢作者的辛勤劳动）。只是可惜对GDI没有一点介绍，还须其他书籍作补充。等搞定这本教程时，应该就已经入了门，可以有针对性的选购书籍，不用担心上当受骗了。还有，VC知识库（[www.vckbase.com](http://www.vckbase.com/)）的网上杂志也很不错，对刚入门的人很有帮助，建议下载。入了些门后，各个网站多转转，看见好文章就往下拉，对提高水平很有好处。   
  
一套MSDN光盘在这时是必不可少的，什么都找中文资料是不可能的，即使能找到也会很费时间和金钱，只能看硬着头皮英文。不过以我的经验看，只要有高中水平的英语，加上金山词霸一路滑过去，看那种专业文章还是很容易的。还有，初学者一上来可能根本不知道上哪找需要的东西，我在这里把常用的地方列一下：   
  
WIN32 API: 平台SDK->Reference->Win32 Functions in Alpha Order   
MFC类库资料: Visual C++ Documentation->Reference->Microsoft Fo...   
C/C++语言及标准库资料: Visual C++ Documentation->Reference->C/C++...   
标准控件: 标准控件都被MFC封装了,可以参考对应的类,如Edit控件对应CEdit类.   
VC中带的ActiveX控件: Visual Basic文档->参考->控件参考(唯一的中文资料,却是最糟糕的。不但是For VB,与C++有很大区别,还翻译得乱七八糟,几乎没法用.如有高手知道哪有英文资料还请告知,不胜感激)   
  
目前市面上的MSDN光盘已经有一段历史了，有些东西可能有些过时。要最新资料的话，可以到微软的网站（当然是全英文的）。不过对于初学者似乎没有必要。   
  
搞定VC后，要学什么都随你便了，再学C++Builder,VB,Dephi,java什么的都是一点难度都没有了，感觉就象是把关键字换了换。或者可以去啃啃COM，据说那是块硬骨头（不过好像也只是入门难，因为有了之前的经验，虽然我没碰过，但花点时间应该是搞得定的。另外DirectX编程经验在此应该是有些帮助的）。   
  
最后，我建议有时间的话多研究一下各种算法。数据结构是一定要看的。不一定要记住（话说回来，不经常用的话也记不住），但至少要知道有那么回事，万一碰到问题了要知道往那边靠。其实，常见的链表等结构以及排序等算法在C++标准类模板中已经实现了，不用你去费劲，但有些了解的总是有好处的。至于离散数学、线性代数什么的，对于复杂些的算法是必需的，其中离散数学是数据结构的基础。不过，我倒觉着只要不是太复杂的问题只用经验也能解决（就象奥赛书中那样，只是实现而没有证明），而且那些东西不用就会忘。万一碰见个难题解决不了，找个高手帮忙也不错，省时省力^\_^。   
  
下面是我想到的一些杂七杂八的问题,没什么条理，凑或看吧：   
  
英语水平不用很高，但至少要有高中水准，否则有你受的。   
  
看书一遍看不懂没关系，事实上，要一遍就能弄懂你就很有天分了。只要多看几遍，综合分析，应该是没有什么问题的。   
  
现在的世道，不是书到用时方恨少，而是书到用时方能学。不要试图满把抓，而要用到什么学什么。这样，学的又快，记的又牢。   
  
一个错误认识－学编程靠的是记忆：这是很要命的思想，会让你累死而无所得。事实上，你几乎不用特意记任何东西。对于各种函数、语句，只要知道有那么个东西，用的时候知道上哪找就行了，用多了自然就记住了。学编程关键在理解编程思想。   
  
一个小技巧：活用编译器可以帮你很多忙。在VC中，当你输入成员符号.或->时,程序会自动列出此类的成员，你只要选一个就行了。这样便省了很多记忆负担。例如，你想获得一个CString类对象的长度，对应方法肯定是Get什么，但是GetLetterNum还是GetStrLength却不知道，你可以输入xxx.Get，这样，你便可从列表中找到一个GetTextLength()，不是这个还能是什么，单击它的话还可能出现说明。还可避免打错字。如果想调用本类的成员函数，就输入this->，就可以查看成员列表了。对于全局函数，则是输入::即可。这样的话看起来还清楚。还有，如果没有出现那个列表，八成是你弄错了什么东西，例如变量没定义，或打错了字，但也可能是VC的BUG,很少就是了。对于初学者，这可是很好用的功能。   
  
初学编程应遵循的一条原则：比猫画虎、不求甚解。关键在于很多东西你就甚解不了，例如MFC中稀奇古怪的宏。而且，甚解了也没太大的好处，暂时还是不要管它，会用就行了，什么时候有了闲工夫再来慢慢琢磨。多留心别人的程序是如何实现的，并照着去做。有时，一些小问题也体现着高深的思想。比如，一个变量要定义在什么地方，如何组织类的结构。如果不太明白，就先找着别人的样子去作，时间一长就会自然而然的明白，这就像下棋一样，老头怎么下，你也跟着怎么下，毕竟他出错的可能比你小。   
  
还有，我觉着初学者不应把过多的精力放在象Cool的工具条，透明的、气泡状的窗口之类的工作上。再搞这些之前，你最好问问自己，自己的程序配得上这些漂亮的界面吗？尤其是，千万不要在刚开始动手时就先打扮界面，以免界面也弄好了，热情也没了。毕竟，看在眼里的成果给人的刺激比枯燥的算法强得多。如果你想搞的话，也要等程序主体定型后再搞。   
  
初学时还应多留心关于程序结构组织的文章，这些东西是很重要的。要从一开始便养成良好的风格。否则有一天当你决定编一个大点的程序时，你就会发现干到一半就干不下去了。   
  
编程不是技术活，而是体力活：学的时候满脑子全是技术，可用的时候就真成了体力活了。真正做起东西来，最重要的是要有毅力坚持到完。意志不坚强就难免半途而废。   
  
关于初学者是该学VC还是该学VB的问题：我觉着，学VC会难一些没错，但掌握学习方法的话也没想象的那么难，而且学VC才能深刻领会面向对象编程的精髓。它使你的思想提升了一个高度，对以后的发展是大有好处的。再说，如果这点困难都克服不了，以后也很难有什么作为。所以，即使你打定主意用VB，也还是花些时间看看VC，至少知道它的大概结构。从长远看，这是很有好处的。   
  
关于学C++ Builder还是Visual C++的问题：也许VC真的不太好，但总不会太糟。可关于VC的资料不知是关于C++ Builder的多少倍。尤其对于初学者，这时很要命的！何况两者相差其实并不太多，学会了一种，再学另一种也不会太费事。   
  
呵，没想到说了这么多。我已经声明了，其实我也就是刚入门，以上一切都是我自己的感受，难免有错误，还请高手指正。另外，这是我第一次写文章，加上我语文很糟糕，其实就是想到哪写到哪，各位就凑或一下吧。   
  
  
  
能有人喜欢这篇文章太好了，我还担心被丢臭鸡蛋呢。其实这篇文章中所有的东西都是我个人的学习体会，所以也难免有些错误。   
既然有人捧场我就厚着脸皮再多说两句：我觉着学编程最重要的个字就是“悟”，VC入门的标准不在于你能做出什么，而在于你是否悟出其中精神。从表面上看就是你是否能做到比较顺利得学下去。至于要怎么“悟”，我就不好说了，好像与人有很大关系。反正我觉着我就比较善于“悟”(^\_^脸皮厚吧)。反正多联系多想就是了。可以确定的一点是，打好基本功很重要。虽说我学VC时间并不长就入了门，C++更是不到一个星期就基本搞定，但实际上我从小学就开始学编程了。到中考后我对C和汇编以及操作系统也有了些模模糊糊的印象。象指针等概念在学习调用中断进行鼠标控制时也有了大概了解。所以中考后没花多长时间便搞定了C++。这之后高一高二一直在学习游戏编程（事实上因为贪玩，大多数时间在玩游戏而不是写游戏^\_^）。在这过程中，我碰到的第一个难题是第一个WIN32程序。不过通过分析每一个变量在何处出现，再加上猜测其可能的运行方式，忽略一些细节，几天后也就能搞一个对话框什么的了，大致了解了WIN32程序的运作。   
  
到了高三，我开始啃MFC（因为身体不好只好休学，只要病好了明年还得参加高考，所以八月份之后恐怕就没时间来这里泡了），不过在这之前我曾次学过一次MFC，不过因为对C＋＋还不太理解而一头雾水。虽说如此，那些模糊的认识还是有很大帮助的，当我花了几天作出一个基于对话框的闹钟时，我便发现我对MFC有些感觉了。（之所以选择对话框就是为了绕开文档／视结构，比较简单）这之后把教程又看了一遍，感觉就清楚多了。然后下载了VC知识库的杂志浏览了几遍（尤其是FAQ部分，有很多小但很重要的知识点），把许多零零散散的知识回想了一下，就感觉基本上入了门了。   
  
以我的经历来看，许多不起眼的小知识点积累起来可以很大的提高你的悟性。所以我建议杂七杂八的文章（即使是VB、Dephi之类）最好能多看一些，毕竟Windows编程有很多是相通的，多了解一些总是好事。   
  
  
另外，有时有问题了只要不太要紧不妨绕过去，不必急于问别人，尤其是基础问题（像怎么画图之类），多找找资料，在许多地方可能都有介绍。而且，时间常常能给你解答。往往一段时间后再一想就会一拍脑门：原来就这么点事呀！。这样对自己是一种锻炼，也节省了别人的时间。   
  
说到学习VC的目的，我并不是太明确。其实以后我恐怕不会从事软件业，很有可能是自动化控制甚至是去当技师（主要是考虑到那时软件业的竞争可能很激烈），所以以后用到VC的可能并不大，只是自己喜欢，并且觉着学习过程就是很好的思维锻炼，现在又没别的事干（考试凑凑或或由大学上就行了），所以就学了。   
  
又罗里罗嗦的说了一堆，让高手笑话了，希望对新手有所帮助。我就权当是练语文了，我可不想明年高考语文不及格^\_^。如果您看着好，就请帮忙转贴一下，先谢了（不过希望能署上史列因这个名字）。

**如何学好VC和MFC（各前辈学习方法及感受整理）（三）**

**如何学习vc++(vc的用处)**   
  
1 vc的用处  
我感觉下面一些领域比较适合于用vc: 操作系统编程，game,图形设计，corba编程,com编程,网络编程。  
我谈谈我的体会。我现在在深圳的一家公司从事IP电话网关的开发。通过这个项目，我 才真正体会到VC有多么强大，上百个线程同时工作，数据库并发访问，而且全部软件基 于COM构造，这样高性能的大型软件，只有VC方显出其英雄本色！不过，在国内，有机会 接触这样的大型系统软件的不多。而在做大型系统软件时，对VC程序员的要求就不仅仅 是熟悉VC、MFC本身了，这只是基本的要求，最的挑战是你如何运用VC实现一些高级的功能特性。  
其实，VC做数据库也不是不合适，关键在于你要抛开所谓的例子程序，直接从底层（SQL）做起。那么，你会有一种天高任鸟飞的感觉：爽。数据库与界面全部在你的控制之下 。  
前一段时间，做一个很大的管理系统（每个纲要540多个表12000多个数据项，纲要数不定），并且还要通过CORBA实现跨平台、通过ODBC跨数据库。如果通过什么PB、BCB、De lphi等等快速开发工具（并且做MIS是所谓强项）将会很难自由发挥的。我以为它比较适合一些需要底层控制的场合，特别是一些使用现有控件不太理想的情况下，VC的自由度就比较高。VC是把全能的瑞士军刀，不过对于一些较简单的功能，比如开酒瓶之类的用个开瓶器就更方便一些。  
VC可以写WEB上的控件。  
VC可以开发优秀的基于通信的程序。  
VC可以开发高效灵活的文件操作程序。  
VC可以开发灵活高效的数据库操作程序。等等等等  
VC是编CAD软件的唯一选择！！！包括AUTOCAD，UG的二次开发。  
小到只是一个Windows的标准界面 那象这样的程序用VC++确实大才小用了，因为有比VC++ 开发界面更快的编译系统,但如果这个界面极其复杂 复杂到要用很多的数学方法（例三维游戏的）那也要用到VC++才能搞定  
就我使用VC++的这几年经验来看，VC++非常适合于系统程序设计，正如C语言是Unix的标  
准系统程序设计语言。  
在多线程、网络通信、分布应用方面，VC++有不可比拟的优势。 当然随着系统程序开发  
的机会越来越少，VC++的应用领域的确会逐渐缩小。  
如果说到Windows下的可视化开发工具现在大家接触得最多的就是VC，VB，DELPHI，BCB  
，一些可视化开发的JAVA。其实我觉得VC并不能完全算是一个可视化工具，这表现在VC  
中编写代码还是占了开发工作的大部分时间。而其他的可视化工具中都在界面设计中耗  
费了大量的开发时间。我一直使用VC的原因就是因为我能够一直将注意力集中在软件功  
能开发上而不是界面设计上。我认为这样能够在开发的过程中更加自由和有更多的控制  
权。而且这种情况下产生的代码维护性更强。举个简单的例子，在维护VB代码时如果没  
有一份详细的说明和流程就会使维护变得不可能，我想其他的基于界面开发的工具都会  
或多或少的产生这样的问题，因为在开发过程中开发工具将一个完整的流程分离成为多  
个部分，在开发完成后这些部分就很难统一起来。  
选择什么样的工具的前提是你的开发目标，如果你希望开发一个很大的系统你就不应该  
选择面向基于界面开发的工具，但你可能会选择VB来开发前端的客户软件，而后台使用  
VC来实现。 对于一些并不是很复杂的软件来讲，界面和操作方式可能是非常重要的，所  
以选择VB，CBC都可以缩短开发时间。这时候选择VC就有些不智。 所以我认为使用VC开  
发的朋友应该将更多的注意力集中在实现软件功能的流程上，多从整体角度看问题。我  
想这一点来说其他的可视化开发工具是很难达到的，因为VB，CBC等开发的程序在很大程  
度上都是用各种控件“堆”出来的，这会在后期的维护升级过程中带来很多的不便，例  
如要替换掉一个控件就可能会对整个程序的结构产生非常大的影响。 最后我想说的是每  
种开发工具都有它的价值，也各有优缺点，更重要的是如何根据具体的任务选择合适的  
工具并利用这些工具来完成工作  
首先要搞清楚VC能干什么.很多人只是听说VC是最好的开发语言，便去学习，就象大家听  
说辟邪剑谱厉害，便都去抢着学一样，都是很盲目的。其实语言并没有好坏之分，我在  
用C之前，一直觉的BASIC 是最好用的语言。现在在WINDOWS平台下编程，VB和DELPHI可  
以满足大多数的应用，而且速度不会很慢。使用VC主要是用来开发系统软件和大型工具  
软件以及开发游戏。  
2 如何学好vc  
2.1  
先期条件：英语要好 看懂MSDN 要学好vc，先学好英语，再去codeguru.com之类的网站  
d/l source code。学好vc=英文好+基本功好+勤奋+聪明。  
1先学习C或C++，在windows的console环境下编写不太深入涉及API的程序；尤其封装，  
继承，多态一定要精 不要让vc的各种特性分散注意力。这一阶段,可以把重心放到研究  
算法上  
2在基本上掌握了语法之后，开始接触简单的系统API，学习 Windows的编程原理和机制  
； 先了解Windows系统的消息机制（多任务，消息映射等）。一开始用基本的SDK形式来  
开发几个程序作为入门(WinSDK编程就是直接调用Windows的API进行编程）(SDK全称是S  
oftware Developers Kit,意思是软件开发工具箱）（API全称是application program  
interface,意思是应用程序编程接口）。开始进行WinSDK编程用C语言最好，可以撇开C  
++的特性专心熟悉一下Win32编程的思路以及了解windows的消息运行机制。可以编写简  
单的菜单程序，可以简单地在WM\_PAINT下操作 GDI函数当你可以熟练的使用C语言进行W  
INDOWS程序设计了，你可以尝试面向对象的方法了。  
3用MFC来开发程序（MFC全称Microsoft Foundation Classes,是把WinAPI进行封装的类  
库，它是一个类的集合，通过覆盖WinAPI，为编程提供了一个面向对象的界面。它使wi  
ndows程序员能够利用C++面象对象的特性进行编程，类似BCB的OWL，Delphi的VCL组件。  
它把那些进行SDK编程时最繁琐的部分提供给程序员，使之专注于功能的实现。你不妨把  
它想象成类似TC提供的函数库吧。）。开始学习MFC，可以从Step tourist学起，看MFC  
的 源代码，理解几个关键的宏的定义与实现，特别是MESSAGE MAP。 在学习的过程中为  
了给自己增加点挑战，尽量不要使用resource edit，试着自己编码实现控件的创建，消  
息的响应。 再就是看自己的造化了，动手做一些小工具，特别是自己平常需要的，对自  
己的提高也应该是很有帮助。 并选择发展的方向. 如果你有一个别人的好程序， 不妨  
自己跟着他的思路，从小到大自己完成一遍。  
4到了一定程度，一定要学ATL，OLE/COM。需要实际项目的磨练研究OLE/COM技术。 COM  
/ActiveX技术是MS的核心技术，只有彻底洞察其理论精髓才可以体会现在的操作系统的  
技术趋势，以不变应万变。  
从1998.3-1998.5，我基本上是每天攻读八小时VC,包括看书和上机。那时正上大三，精  
力充沛，受的起没完没了的失败和痛苦，终于在五月某一天的一节“汇编语言”课上突  
然想通了，立刻逃课去上机。就这样我入门了。之后我也象大多数的人一样学习它的所  
谓GUI,doc/view结构,打印之类。有一天醒来我突然想：我学这些干嘛吗？朋友！惊醒呀  
！人的精力是有限的，正所谓“术业有专攻”，VC说到底，它也是一个工具，学会使用  
这个工具并不是目的，而仅仅是开始！于是我开始寻找我的方向。综合我的方向有：1.  
网络编程。主要是基于TCP/IP的Winsock编程，已完成一个FTP的客户端编写,正在写一个  
FTP服务器。2.通讯编程，主要是串行通讯。3.三维动画，主要是用OpenGL API，一般用  
来写界面。我的毕业设计就是以2和3为基础。我的一些意见：不要对VC控件抱太大希望  
，除了基本控件，使用其他控件会使你比用API还繁；尽量使用MFC而不是API；尽早定位  
自己，寻找方向，不要什么都学，人贵专攻；要了解消息机制，多线程和同步机制，更  
有回叫（CallBack)机制等，这是VC程序员的基本知识  
//WINDOWS 和 UNIX操作系统都是复杂的多任物系统，提供了一大堆应用编程接口（API  
，通常是C语言的函数）程序员使用c语言调用这些API便可以开发该系统下//的应用程序  
了。这与dos时代的编程接口是不一样的，那时侯的函数库是由开发环境提供的（如Tur  
bo C) ，不具有很好的封装性和设备无关性。每当新版的//WINDOWS操作系统发行，便会  
提供一个相应的plantform SDK(软件开发包），开发者可以用SDK 编译C程序。  
//vc的核心就是MFC，MFC是个C++类库，就象结构化程序设计时代的C语言函数库一样 ，  
给程序员提供了丰富的编程接口，简化了程序的设计。 而MFC就是直接把//WINDOWS的C  
语言编程接口API函数用C++的类封装而成！ 这样既实现了面向对象的编程思想，又直接  
使用了WINDOWS的原始编程接口，代码的效率是 最高的  
//1。Window是由个许多DLL搭起来的系统的总称，开发Window程序就是如何调用这些DL  
L，对于C程序员，它提供了.h/.lib,这些东西和一些文档就构成SDK,DDK。//。。他们在  
任何支持DLL调用的开发环境中都可以使用（即函数指针）。2。通常SDK指与Window核心  
调用有关的。。。但并不是专用名称。也用于其他地方  
//VC只是一个编程工具，就如BC、BCB、Delphi一样，其实对于编程最根本的就是三个方  
面，语言、开发包、操作系统API，他们三个方面应该说是相对比较独立的//。VC是MS开  
发的，所以针对的是windows api，你可以不会C++，也可以在VC下用C写出很优秀的程序  
，当然如果你比较熟练C++，并且熟悉开发包MFC的话，工作//可能要轻松不少。所以在  
我看来，学习的过程可以是这样的，1、先学习C或C++，在windows的console环境下编写  
不太深入涉及API的程序；2、在基本上掌握了//语法之后，开始接触简单的系统API，学  
习Windows的编程原理和机制；3、在可以编写简单的菜单程序，可以简单地在WM\_PAINT  
下操作GDI函数后，开始学习MFC//，可以从Step tourist学起，看MFC的源代码，理解几  
个关键的宏的定义与实现，特别是MESSAGE MAP。在学习的过程中为了给自己增加点挑战  
，尽量不要使用//resourceedit，试着自己编码实现控件的创建，消息的响应。再就是  
看自己的造化了，动手做一些小工具，特别是自己平常需要的，对自己的提高也应该是  
很有帮助。

**如何学好VC和MFC（各前辈学习方法及感受整理）（四）**

学好VC++的十大良好习惯  
  
每到年底各大媒体就争先恐后热火朝天地搞总结，什么十大人物，十大品牌，十大美女，十大帅哥等等五花八门乱七八糟的让人充满好奇充满怀疑，这事确实让人有点郁闷，就如同男足国家队的国产教练如沈墙扶们每一次踢球失败后都要说这么一句：我们回去后要好好总结，下次会打得更好！　这话听了几十年了，耳朵都生虫了，但还是无法看到中国猪球队有象人样的表现。因此，总结在某一程度上来说只不过是一种形式罢了，总结不代表就能改过原有的不足，也不代表就能进步了，甚至有点俗不可耐，尽管如此，阿蒙亦明知故俗，前人说过了入乡了就要随俗，因此你生活在这种环境里，你无法对这些无聊无趣的东东置之不理，除非你是天才，天才往往在非天才的人看来是很怪异的，处处与现实格格不入，可阿蒙不是天才，所以还得赶快总结，要不就离题，又被大家骂了，：）   
  
  
  
　　（一）充分利用MSDN，因为我个人觉得它胜过任何一本编程参考书；   
  
　　MSDN是　Microsoft　当前提供的有关编程信息的最全面的资源，它包含微软最新的技术数据库，加上易学易用的全文检索功能，让您迅速找到任何您需要的技术参考数据，让您随时拥有与全世界菁英同步的技术，掌握最丰富的程序开发资源。我经常收到很多朋友的EMAILS，他们所提的问题往往都非常的简单，MSDN完全可以解答这些问题，但他们好象不太喜欢用，这是让我郁闷的地方，是因为英文不好呢，还是没有学会充分利用各种资源来解决问题的方法呢？   
  
　　（二）提高英文水平，养成多上英文网站多看英文资料多买老外原版英文书；   
  
　　有关程序员与英文水平的讨论已太多太多，　我个人认为要成为程序员，高中的英语水平够了，甚至不懂英语的一些人，也同样可以成为较好的程序员，因为开发工具的发展将是越来越傻瓜，但如果你是仅仅满足于能运用某种工具开发某个软件模块，那是没话说了。真正热衷技术肯干钻研乐于接受挑战的程序员是不满足于现状的，他们总感觉有太多的未知，于是总在不停地学习，如今信息技术发展得太快，而大部分的技术最先出现的时候都是英文版本的，要几个月或者几年以后才有中文版本的书出来，因此要想跟上步伐，一定要努力提高自己的英文水平，这样才能同步跟上信息技术。你可能担心自己的英语水平不行，没关系，刚开始多查字典，“万事开头难”，必须有持之以恒的精神，不久你就会发现计算机英语其实很容易的。何况很多   
  
　　英文技术站点确实比国内做得好啊！比如http：//www.codeguru.com，http：//www.codeproject.com，http：//www.programmersheaven.com　等等。   
  
　　（三）加强自我管理，善于作自我总结，分析自已的优点及缺点。   
  
　　中国境内百分之八十以上的领导人在百分之八十以上的场合的讲话中都有类似的观点，所以在这里我是不多说了，反正这一条用在什么行业什么地方都不会有错的，人生最大的敌人不是就是自已吗？管好自已认清自已，那还有什么搞不定的？   
  
　　（四）养成良好的文档习惯   
  
　　程序员大多都不喜欢写文档，我以前也是特讨厌，在我的思想里，所谓的文档就是一些废话，一句话硬是用十句话来代替的无聊透顶，就如同部分中文系男生的爱情表白，明明就是“我爱你”三个字，他硬是把月亮啊太阳啊大海啊高山啊石头啊天使啊乱七八糟的都拉上关系了，尽管听起来浪漫，但在我认为不实用，：），　甚至太肉麻了，一个男子汉干嘛这么罗里罗嗦的……良好的文档是正规研发流程中非常重要的环节，一个好的程序是先写好设计文档再进行编程的，在设计文档的指导下，才能写出安全的代码。如果你不写文档，一开始就写程序，这样你就不会按已设计好的路线走，而是想到哪写到哪。小功能还好说，要是大功能，就容易混乱甚至失控。那么如何写文档呢？其实我认为没有统一的标准，虽然国家及一些NB的人总结了很多的模板，但每个人的习惯不同，如果你不加以修改或创新，就套用某个标准，我相信写起来会很吃力及说不清的难受，因此我觉得只要能将你的设计思想及实现算法或步骤描述清楚就是好的文档，我强烈建议广大程序员朋友们在写文档时要善于用图表来说明你的思想，我们不是作家，也可能作文都经常性地不及格，写出五官端正的文章对我们来说可能不容易啊！好好地利用VISIO，ROSE或别的工具来表达你的思想吧！   
  
  
（五）代码风格要规范，严谨，效率要高。   
  
　　这个不用说了，所以一定要记住了！不过，这一点有时可能与人的性格有关，如果你是经常丢三落四经常胡子长长经常钮扣扣错经常吃个快餐要一个小时的人，那你在CODING的时候可千万要注意了，CODING是CODING，生活是生活，不要写出的程序也是那样就不好了！   
  
　　（六）掌握好跟踪调试技巧。   
  
　　跟踪调试程序是一件繁琐而又复杂的事情，所以掌握必要的调试策略及技巧却可以使这些工作变得轻松起来。强烈建议你去看一下老美Everett N。McKay及Mike Wooding写的书＜＜Debugging Windows Programs＞＞，你一不定受益匪浅。   
  
　　（七）养成自我测试的习惯   
  
　　测试工作应由测试工程师来做，但在你写完一个模块或一个软件时，还是要自已先测试一下，保证不要出现一些低级的错误，何况这些错误让测试工程师看到了，狂扁你一顿，你很没FACES的。   
  
　　（八）善于交流善于沟通，特别是经常与一些高手交流一下学习的心得体会；   
  
　　有人说，程序员的性格大多内向不喜欢说话，其实是有些误会了，不是不喜欢而是话不投机，我的脑袋一天到晚都在不停地转，函数，数据，算法啊充满了我的世界，我那还有时间与你谈一些无聊的话题，话要找对人了，才容易谈下去，书上说过“听君一席话，胜读十年书”，你要找的就是这种豁然开朗！现在技术的论坛越来越来，这将成为程序员交流一个重要的地方，也有人说：“读君一长贴，胜读十年书”，：）   
  
　　（九）阶段性地做一下专题总结   
  
　　知识要温故而知新，因此我建议程序员要养成阶段性地做专题总结的习惯，比如你这个月学习或在做与多线程有关的模块或项目，那么在你做完后，你就可以好好地总结一下所有与多线程相关的技术，包括理论知识，实践方法以及各种技巧及优秀文章等等，这对你各种能力的提高将有很大的帮助，你试过了吗，如果没有，那就快点行动吧！   
  
　　（十）要有持之以恒的精神   
  
　　这是废话，因为我揍不齐十大，所以将它也算上，中国自古以来喜欢号召大众学习某种精神，比如马克思的，列宁的，毛泽东的，邓小平的，雷峰的等，这些精神使社会更安定人民生活更美好，那么程序员要有什么样的精神呢？我不是我说了就算了的，我只是想说明要学好任何一门技术，最好要有持之以恒精益求精的精神，特别是学一些比较抽象比较难的技术，比如VC＋＋，我想它应比别的开发语言都要难学些，或许你已经开始了两年了，但感觉还是不爽仿佛也没掌握什么，这个时候你除了思考一下你的学习方法以外，还必须坚定你的目标及信念！

**如何学好VC和MFC（各前辈学习方法及感受整理）（五）**

如何成为一名优秀的程序员?  
  
  
一位仁兄说的“程序员写的程序不是算法+语法 ，而是要能够满足用户需求的工 具”我非常赞同，要想达到用户需求就必须从各个方面来考虑如业务、人机交互 、效率等方面，而不只是一个语言（语法）的问题，语言（语法）只是工具，只 知语法不知其他那就真是编程机器了！ 编程机器在印度高中生经过几个月培训，按照严谨的软工方法，加上较高的管理 ，就可以胜任了！大家相信吗，我是相信的！谈到这里我就不禁说到了国内教育 界最近在探讨的问题“计算机科系的毕业生特别是本科大专生到底出来干啥、如 何适应社会要求”，大家也看到了很多计科系大学生说精通N种语言，熟悉N种工 具，不知道学校里的其他知识到那里去了，甘愿做编程机器，浪费了人民的纳税 ，干高中生能干的事，比较可惜吧！在国内现在就是这样了，看过一则帖子：清华的计科系毕业声在建筑院里搞开发还不如建筑专业的毕业生。说着说着就岔道 了，国内的软件开发业到底是需要那些人：如果仅仅是编码机器，那我估计中国 硅谷还是做梦去吧！  
社会似乎也需要编码机器，翻翻招聘广告，做应用开发的都要求精通某某语言， 熟悉某某工具，很少需要懂管理懂软工的人。以我个人一点偏激的想法，民族软 件产业要腾飞，更需要的是能管理使用编码机器的人，即管理人员、国内软件产 业编码机器已经很多了。希望不要惹怒了那些编程高手！ system develop与Application develop在国内到底哪个能养活你，能赚钱，诸位 仁兄想必也知道，况且俺也没发现几家水平高的公司招这方面的人，毕竟OS，DB MS，COMPILER都被国外做了、另外也别跟我谈LINUX，毕竟还是少数烧钱的人做的 事情，我先喂饱肚皮再说。我手下的很多搞4GL语言的程序员都想转行学VC等所谓 的更低级的语言，我总是说“在XX城市，先用4GL工具生存，以后再学习VC吧！” ，说的简单一点先解决肚子问题。如何判断自己是否是编程机器？   
1、面对需求不考虑用户，只是考虑用那些程序技术展示自己的语言语法技巧 .  
2、学习了N种语言 .  
3、从来不学习或实践软工 .  
4、语法语言水平在众人中遥遥领先、特别是一些稀奇古怪的语法 凭着兴趣和创造力去干，却重复繁琐的劳动。 做着没有意义-唯一意义是赚钱，而且真是出了半斤力， 拿不足八两。 终日劳累，却不能学自己想学的。最终结果是跟不上社会科技的发展 ，人已衰老。悲哀！！！   
　  
  
开发软件的关键是要有想法，一个好的想法比什么都重要。尤其是有关 网络方面的就更是如此。   
  
入门还可以，但是要继续深入了解可能要难点。 未必吧 偶觉得大学里的高数 数理方法之类的, 如果你不是做研究的话, 应该是很少能 用到的 不过如果说到离散之类的, 倒还是时不时的能有点用现在的程序员比起十年前是不是要花更多的时间来 查帮助呢。系统越来越大，手册越来越厚，软件开发的 周期是不是越来越多的淹没在查帮助之中了呢。   
  
实际上一个程序员最终的技术需要和实际相结合。真正在编写程序到达一定时候，语言的使用并不是最大的障碍，对整个项目的把握、软件工程的把握、数据库的设计以及执行效果的分析等等才是需要进一步考虑的东东！否则，为何大多数公司到要求有编程经验了！这些不是程序员必须学的。但数据结构，编译原理，操作系统原理等是必须要学好的，英文的多看，不懂计算机英语可不行。实我不是什么中专生,而是我读的中学和一个私人办的电脑学校联合开的电脑专业(并不是我中考考的差,而是这个学校太贪钱了,才被录取到这里,恼火,我们班里中考成绩从两百多到四百多的都有,我就是四百多分(重理轻文的结果,要不然...),当时读书的时候,我是班是的高手(其实只是比其它同学懂而已),大家叫我dos,因为当时学的都是dos的内容, 毕业后还没有对编程很感兴趣,只想找一个电脑的工作就可以了,哪怕是打字的,可是看报纸,去人才中心,看到都是要大学的,为此感到很失望.也对电脑失去了兴趣.后来学校打电话到我家,说厦门厦华公司要招工,要不要去,我很快就答应了,因为当时没工作,天天呆在家里.后来打工的时候,天天象一个机械人一样,重复着同一道工序,因此经常在深夜的时候,思索着自己的未来,由此重新生起了对电脑的感情,因此经常买电脑杂志和报纸看(可以堆成一座山了).由于离我住的地方不远处,有一个电脑培训的,所以经常到那里上机,而在学校里学过的软件也只有FOXBASE和WPS,其它的不值一提,所以上机经常用foxbase,直到这时候,才对编程产生了浓厚的兴趣,一年后,自已买了一台电脑,开始认真学习编程.由于我是属于职业中专的,因此经常想,就算学得再好也没有用,所以想考程序员,而考程序员要懂得c语言,所以就学习turbo c,学完了,学数据结构,同时看'C高级实用程序设计',澳门回归和2000年的两个晚上,我都是在编程中度过的.由于我这个人对书很感兴趣,经常在星期六,星期七去书店,而在书店里,也是看编程方面的书,而看到的编程书籍大部份都是windows方面的,为此也经常思索着学dos编程到底有没有用.后来,春节放假(要2月13日上班)回老家,天天去新华书店(正月初一也去),看到也都是windows编程方面的书多得像狗屎一样,所以就下决心学windows编程,因此正月初四(快餐店还没有开张)就去厦门了,很快买了delphi的书和d版delphi5,疯狂地学习(到目前为止,买了8本delphi的书,因为国人的写的书实在太烂),而由于遇到不懂的又不知道怎么办,为此想到了网络,但在外打工不可能上网(上网吧太贵),所以就辞职了(4月21日),现在,程序员考试快到了,是报还是不报一直犹豫不觉,困此才有此问题.打工的岁夜,我永远不会忘记,因为付出太多了,也失去了太多(坏了两个光驱,瘦了几斤),直到现在,脑海里还不时浮现起那几个无眠的夜晚.忠心感谢大家.我不认为编码的人就是机器, 而系统分析就不是机器, 其实系统分析员就是销售的机器, 所有职员有是老板的机器.它们之间这是不同工种吧了, 当然对系统分析要求要高一些, 薪水也高一些, 但更让人佩服的是销售, 是他们驱动了整个的运作.我也是个中专生, 还是学机械的(后来自学了计算机), 我非常了解在传统的制造业是如何的规范, 设计人员设计图纸, 然后经审核, 再到车间试样, 再根据情况, 修改图纸, 如此反复几次后才能一个产品定型,而在软件界, 就没那么好了, 领导会说, 这个你做, 那个他做, 也没有经过很细的分析(国内很多都是这样), 在我们这里也没有系统分析员, 每个人都是设计员, 也是编程员, 虽然这样对个人来说, 能学到很多东西, 但不利于项目.我国的软件过程水平,确实令人担忧, 目前为止只是, 几个人十几个人的小软件, 还没有能拿得手的大型软件.至于中专生编程问题, 我  
认为只要入了这个行, 就不会比本科生差, 因为对他来说没有优越的学历条件, 那么只好埋头苦学, 但这正好适应当前计算机软件迅速发展的今天, 学历只能代表过去和基础, 更需要的是有能力的人, 解决问题的人, 实干的人.对我来说确实有时有点自卑, 没有上过高中和大学(由于那个年代, 我只知道能为父母减少一点负担就行了), 所有我一直在努力的学习(corba, uml, java, 软件过程等), 目前为止我并没有觉的我的构架能力和编码水平比他们差, 只是觉得E语言实在太差.过计算机本科又如何？我有几个同学到银行去搞业务了，有同学任教，有同学收税去了......60多个人真正现在搞计算机的还就只有几个人，再看看当时这些计算机本科生的毕业设计,//faint有的人到最后连vb都搞不定，但他们什么编译原理啦什么组成原理啦什么软件工程啦学的（应该是考试的分数）真还不  
错，至少我感觉有些概念比我清楚（上课没听?)所以我觉的中专生并不比一般大学生差（当然有些重点学校除外//hehe)，有时中专生有更大的压力逼迫自己去学习，学历不是重要的，关键是一个人的素质.我们不能将目标定位在做程序员或编码员（Coder)上光会写代码有什么用？那叫“编码员”，在国外是属于体力劳动的，不像国内，会写程序的就叫高科技。重要的是分析问题、解决问题和规划的能力，系统分析，系统设计及项目规划才是正途。这就需要学习所谓的基础课程了，如：软件工程、离散数学、数据结构等等。从vb到现在开始学vc后，一个人捣鼓了几天也没有什么新发现。跟本不知道VC的编程思想是什么，请大虾们告诉我，它和VB的差别真是太大了。VC的博大在于MFC的操纵,它是Win32API的封装.思想在于怎样了解MFC的内幕,它的运行机制.差别也大，差别也不大！这要看你对API的理解了，虽然VC++的可视化没有VB的好，但是不是绝对没有的，其对而且对话框的编辑是跟VB一样的，不过不是像VB那样放在第一个界面罢了，VC++的博大精深是VB难以望其项背的！：）而且VC++是完全面向对象的编程工具，而  
VB是不够完全的面向对象编程工具，VC++是完全编译语言，VB是本地编译语言，不够完全，VC++效率高，封装性好，继承性高，VB效率相对低了很多，但界面友好，二者只能取其一，或者使用VC++，VB辅助(因为VB开发快)，当然Delphi，BCB也是不错的选择。

程序员不应依赖开发工具，程序员更应该拥有的是一种思维、一种精神、一种观念。就像Richard.M.Stallman一样，有自己的精神，为自由软件而奋斗。就像求伯君，为民族软件的振兴而奋斗。这才是真正的程序员。  
应该说，他们更注重的不是技术，而是软件的思维，软件的灵魂！！我刚学ＶＣ的时候,还没有上网.身边也没有一个可以问的朋友,所以大部份都是自己啃的.那种感觉真是很痛苦. 现在在网上就不同了,可以得到太多的资料了,而且还可以得到在线帮助.但这些都不是学习的关键. 相信各位也知道ＶＣ的难度,并不是那么容易上手的,所以要想学会,学好ＶＣ,靠外力是不可能的.主要得靠自己. 自己要有一份难得的毅力,对编程的狂热也可以在一定程序上起到帮助.我就是这样的.起初,没有人帮我,我学习ＶＣ是三天打鱼两天晒网,学习进度很慢,幸好对编程的执著,使得自己坚持下来了. 如果你从来就没有接触过编程,那你学习ＶＣ的速度可能会比学过面向过程编程的人要慢一些,因为你要去理解命令及语句的含义.但只要你努力,并且可以得到别人的帮助,我相信在半年内会对ＶＣ有一定的认识. 请学赤面向过程编程的朋友也不要笑,因为面向过程与面向对象实在是区别太大了.就拿封装一词来说吧.当初我是左想右想才想通的.所以不要自己学过编程,就会在学习ＶＣ的通道上比别人轻松. 现在有一种现状应该让我们注意.我发现有很大一部份初学者觉得ＶＣ是一种语言,Ｃ++又是另一各语言.我在和一些初学者的交谈当中,查觉到了这一点.有的初学者竟然还认为我学ＶＣ为什么就一定要学Ｃ++?我想这个问题是我们大家都没有注意到的一个问题.就是向初学者讲述Ｃ++对ＶＣ学习的重要性. 我这有个例子,跟大家讲一下. 我有个同学,他接触编程比我要早,在我还在为ＶＣ中"::"符号怎么标记的时候,他已经在学习Ｃ了.后来,我对ＶＣ稍有理解的时候,他也发觉Ｃ的跟不上时代的脚步了.我便提议他从Ｃ++语言学起,可他认为自己有Ｃ的学习功底,根本就用不着再去学习Ｃ++.可在学习ＶＣ的当中,遇到的困难真是数不胜数.最近,他还是去买了一本学习Ｃ++的书.从头再来学习Ｃ++. 我希望通过这个例子,能让广大的初学者知道,Ｃ++对ＶＣ学习的重要性。  
学习VC必须有狂热的编程热情，否则是很难坚持下来的，我周围就有几个这样的人，他们比我先学VC近半年，但现在仍然学不会，而我现在虽说不是很厉害，但基本的应用程序是不在话下，我就是天天看书，上机实践，几乎所有的时间都泡在里面，有时连吃饭都在想，为什么，因为我确实想啃下这块硬骨头，我不想半途而废，我觉得学习VC不仅仅是学到了更多的东西，最主要的是培养了我们自己一种坚持克服困难的毅力。  
对于VC，我有几点经验：   
1。技术为本，语言为次.   
2。MFC的单个类有用，DOC/VIEW要小心。   
3。OOP要小心，使用不当反而造成大量的工作和糟糕的代码。   
4。如果可能，考虑选择使用Delphi(CBuilder+VCL)。   
5。到了一定程度，一定要学COM。   
要学VC，必须有对C++深刻的理解，对WINDOWS运行机制的深刻理解。尤其想成为VC高手。举例 ， 对于虚函数，不仅要知道有这么一个东东，更要知道它的内存镜像 。这样才能对VC中很多的东西举一反三，事半功倍。本人学习VC近2年，但自觉第一年由于心情浮躁，把VC的书翻了一遍又一遍，却每次都只看了一点就无法再深入。直到毕业前夕，痛定思痛，克服浮躁，认认真真的从最基本的开始学，把每一点都搞的水落石出，经过三个月的刻苦，终于大成。直到现在，半月搞定COM/DCOM，都托当日刻苦之福，因此劝告想学VC的朋友，一定要顶住开始的艰难岁月。成功属于刻苦者。  
  
每个人都有自己的学习方法,也许这种方法对我来说有用,但不见 得就对所有的人有用.所以,请不要盲目的跟着别人的学习方法学习,要思考属于自己的学习方法. 但我还是会向大家说出我自己认为比较好的一种方法. 学习编程其实与学习其它东西一样,要想掌握它,就要实践,实践,再实践.当你学到了一种新的技术或知识时,多实践是巩固学习的一种最好最有效的方法. 这个实践不是照著书上的例子做一遍,而是根据自己的能力,给自己出题,然后去完成它.只有这样,你才能发现自己的不足,同时又增加了自己的编程经验. 但要成为合格的程序员,光  
会写代码是远远不够的,更重要的是思考.谋定而后动,是 不变的真理.   
在我的理解，VC只是一个编程工具，就如BC、BCB、Delphi 一样，其实对于编程最根本的就是三个方面，语言、开发包、 操作系统API，他们三个方面应该说是相对比较独立的。   
  
VC是MS开发的，所以针对的是windows api，你可以不会C++， 也可以在VC下用C写出很优秀的程序，当然如果你比较熟练C++， 并且熟悉开发包MFC的话，工作可能要轻松不少。所以在我看来，学习的过程可以是这样的：   
1、先学习C或C++，在windows的console环境下编写不太深入涉及API的程序；   
2、在基本上掌握了语法之后，开始接触简单的系统API，学习 Windows的编程原理和机制；   
3、在可以编写简单的菜单程序，可以简单地在WM\_PAINT下操作 GDI函数后，开始学习MFC，可以从Step tourist学起，看MFC的源代码，理解几个关键的宏的定义与实现，特别是MESSAGE MAP。 在学习的过程中为了给自己增加点挑战，尽量不要使用resource edit，试着自己编码实现控件的创建，消息的响应。 再就是看自己的造化了，动手做一些小工具，特别是自己平常需要的，对自己的提高也应该是很有帮助。 究竟我们要的是结果！良好的分析问题高效清晰地肢解问题的能力才是我们真正要不断学习的吧？  
和学习其他知识一样，重要的是获得提出问题，分析问题，解决问题的能力，不是为编程而学习，你具有什么样的思想，就会写出什么样的程序。学一门语言，不能仅仅是语言，要注重语言背后的思想方法，看他是如何来解决某一问题的，为什么要这样去做，他总是要符合客观事实的，就像人说的话一样，存在某种逻辑，数据的组织，信息的传递，靠你自己的头脑去建立，然后看C++中有什么可以帮你的，该怎么样用他来表达自己的想法。只要你认真实践，努力去做，寻求乐趣，就会达到目标。我虽然 真正认真学习编程的时间不长，但现在我是认真的，就有了以上的体会。  
  
学习VC++有相当的内容要学，而最后的深度则看个人的悟性与勤奋了。   
1）掌握最新标准的C++。(2个月)   
如果曾经在大学里自以为学过C/C++,还对dynamic\_cast/static\_cast/template/try/catch/throw/stl/...感到  
陌生，那你该Refresh一下新的ANSI C++标准了。 ----此与VC++无甚关联，g++/bcb均有所支持。  
2） 学习SDK编程；:-O （6个月） 痛苦是暂时的，必要的，坚持就是胜利！ 建议用Lccwin32/Masm32/Tasm编写小而精的工具软件； "Windows Programming Guide." "Advanced Windows programming "   
3） 研究MFC源代码。(6个月)   
不要被一大堆的宏所蒙蔽，just track and dig into it!! 在知道MFC如何扩展，包装SDK之后，自可功力渐进，不被其MFC系统框架所困绕。"MFC Internals"   
4） 研究OLE/COM技术。   
COM/ActiveX技术是MS的核心技术，只有彻底洞察其理论精髓才可以体会现在的  
操作系统的技术趋势，以不变应万变。   
"Inside Ole 2"   
  
\*\*\*几点建议\*\*\*：   
1) Delphi/BCB/VB可以很快构筑界面，但对于想真正学习、理解系统不是一个好的平台，但如果有时间读一读VCL的源码，看看Borland是如何封装系统的，也可借鉴不少。   
2) 学习ASM对理解C++有很大益处。Soft-ICE也是和VC++结合在一起的好工具;   
3) 语言本身是皮毛，算法是筋骨；无论高级语言如何发展，在理解操作系统的基础上不断自我创新的能力是源源不断的；如果精髓一旦了然于胸，选择何种开发工具都可以驾御，一见如故了。  
  
每个人学习的过程都会不同吧，我在98从TC转向VC时，对Windows的消息机制一点概念也没有，对着个MFC不知所措，几个月下来一点进展也没有。我于是暂时转向win32位编程。整整一年时间吧，我就是与API打交道，连编辑菜单条对话框等资源文件时也用Uedit32手工编写，为的是希望能对编译器的全过程有个感性的了解。之后我再转向MFC时，果然发现两者是相通的，虽然有一定的区加，不过有了win32位编程的基础再看MFC 时会发现它比win32位编程要方便了太多。 现在市面上的VC书很多，不过大多是入门书。我以为要精通VC（只是以为，本人自已距精通还远的很）应该多看多写程序，看书只能是入门，会用了而已。我不赞成滥用ActiveX，一来我以为它的性能很成问题，真是又大又慢又不稳定（可能是我有点偏激了吧），二来用了它您的程序今后就要被别人牵着鼻子走了。有次我用了个控件，程序都写了大半了，它给我来个继续使用请支付333美元，我两个多月的工钱，烦不烦人？现在我尽可能用别人写的类源代码(我已收集各类代码3-400MB了)，一样用起来方便，还能边用边学，必要时还能自已改造。 我以为在现代的网络时代，资料到处都有，无论什么先进的技术，今天我不会我只要查到资料，快的学上几天慢的一两个月我也能学会，现在已没有写不出的程序了，写不出是因为你没有耐心写完它。我以为现在的程序员不一定要多么聪明，而更需要他有完成枯燥项目的耐心，找BUG的细心，对于金钱的平常心  
（不要为了早日拿到钱而赶进度），最后最重要是有顾全大局，团队协作的精神。 最后，王靖朋友的经历与我实在是有点相似，算是同病相伶吧，真心祝您早日实现自已的人生目标！学习VC是一项费时费力的艰苦学习过程。为了真正用好VC，你 不得不先从OOP开始学起，也许浏览一本纯理论的书要更有意思。 你将从另外的角度考察OO思想。接下来学习C++，学习WindowsAPI 学习MFC，剖析MFC，扩充MFC，创造自己的类库（不要企图写一个 和MFC同重量级的类库）。如果能够精通Microsoft C编译器的各种 参数，你就可以开始研究微软C/C++语言编译器的进化历史（结合 各种背景知识）。 掌握了编程的思想，征服VC只是时间问题。  
  
学习VC是接受微软技术体系的过程。所需要学习的不仅仅是C++，MFC。 需要学习所有微软的技术，包括Windows编程，Win32系统（实际上 Win32系统实现了很多我们在屏幕上能看到的东西，最起码NT的内核 并不大），COM（深入研究它，理解对象是如何跨越进程边界的，最终 能够自如地在Exe中实现自己定义的接口才算到家了。不过这多少有些 不太必要。），DCOM，ctiveX，Windows DNA... 如果你想成为大拿，学吧，没完没了。最起码，使用VC，你甚至于可以 开发自己的操作系统（当然与MFC没有一点关系）。