电脑编程技巧与维护月刊

圆圆圆年第 远期　　总第 苑圆期　灵怨源年 苑月创刊）　　每月 猿日出版

名 誉 社 长　张　琪
社　　长　孙茹萍
副 社 长　毕研元
总　　编　王路敬
执 行 主 编　叶　永
副 主 编　杨林涛
编　　辑　程　芳　管逸群
　　张　涛　司维清
公关部主任　黄德琏
出版发行部　毕　波
编 辑 出 版　《电脑编程技巧与维护》杂志社
法 律 顾 问　佟秋平 宋亿中律师事务所 雪
主 办 单 位　中国信息产业商会
社　　址　北京海淀区学院南路 远怨号吉安大厦 源苑室
投 稿 信 箱
编辑部信箱
邮　　编　员圆园园圆员
电　　话　园员园-缘圆圆圆苑圆园园
传　　真　园员园-缘圆圆圆苑圆园园
照　　排　《电脑编程技巧与维护》杂志社电脑排版部
印　　刷　北京巨龙印刷厂
订 阅 处　全国各地邮电局
国内总发行　北京报刊发行局
邮 发 代 号　员圆-苑缘
刊　　号　ISSN 员园园圆-园园园圆/TP·怨圆园
　　CN 员员-猿缘园园/TP
广告许可证　京海工商广字 圆缘苑号
全 年 定 价　怨缘圆元
每 期 定 价　苑缘元



新技术追踪

猿　希捷公司推出新版 阅酝蕴都宰窑耘酝硬盘安装软件等员怨篇

编程与应用起步

缘　使用 阅灶云创建程序组和程序项
苑　灾月中与通讯口交互的实现方法
怨　创建 宰窑耘酝怨载蕴愿快捷方式技巧
员园　在 岳蕴耘酝载下对文件与目录进行编程
员员　在 灾窑造云悦则中提高程序性能的技巧
员圆　网页中多层效果的灵活使用
员猿　元宰窑耘酝载愿下的多显示器编程技术
员源　一个多路循环数据采集系统的实例

编程语言

员缘　悦垣垣与 云则则混合语言编程中动态链接库的调用
员苑　利用 阅蕴窑耘封装 配则窑耘实现网络实时数据广播
员怨　配则耘文件的编写

专家论坛

猿员　酝云悦中内存泄漏的检测
猿源　宰窑耘载中取得 员环

需要全本请访问 www.ertongbook.com

《电脑编程技巧与维护》杂志社授权佟秋平律师声明

《电脑编程技巧与维护》杂志社出版的《电脑编程技巧与维护》期刊，以荟萃国内外电子计算机先进信息，以领先性、实用性而享誉电脑报刊市场。但是近来《电脑编程技巧与维护》杂志社接到许多读者询问：“重庆科潮电子图书有限公司”在全国多种报刊上刊登邮购本社杂志 [illegible] 年合订本的广告。已有一些读者将定购款汇至“重庆科潮电子图书有限公司”后却未得到本社 [illegible] 年合订本，且与“重庆科潮电子图书有限公司”无法联系。“重庆科潮电子图书有限公司”的行为严重影响了《电脑编程技巧与维护》杂志社的声誉，也损害了读者的权益。为此，本律师郑重声明：

《电脑编程技巧与维护》杂志社的发行和销售渠道是报刊发行局和各地新华书店及部分代理书店。《电脑编程技巧与维护》杂志社并未授权“重庆科潮电子图书有限公司”刊登邮购广告。敬告广大读者如需订阅《电脑编程技巧与维护》杂志可向上述代销商订购，也可直接向本杂志社订购。希望广大读者提高警惕，辨清真伪。

特此声明

《电脑编程技巧与维护》杂志社法律顾问：佟秋平律师

[illegible] 年 [illegible] 月 [illegible] 日

控制权的方法

新一代用户界面技术 原创杂[illegible]技术及编程介绍

可视化专栏

用 [illegible] 开发一个 [illegible] 的增强版—— 兼谈 [illegible] 的 [illegible] 机制

利用资源文件编写安装程序

[illegible] 系统中数据的直接输出方法

自动列表技术在 [illegible] 中的实践

关于隐藏 [illegible] 任务栏中的应用程序按钮的几种方法的介绍

数据库

找回 [illegible] 数据库中忘记的密码

适应不同开发工具的 [illegible] 设计与调用

网络技术

非阻塞 [illegible] 编程的问题及解决办法

定制自己 [illegible] 的一般方法

用 [illegible] 制作拨号上网组件

基于 [illegible] 的 [illegible] 聊天室系统

图形图像处理

图片的平滑切换处理技术

游戏开发之路——在 [illegible] 中进行 [illegible] 编程（七）

计算机维护

在 [illegible] 图形中标注坐标

源码 [illegible] 编程实现 [illegible] 文件“减肥”批处理

计算机安全

设置可靠的防火墙系统保护公司内部网络

用 [illegible] 服务器和 [illegible] 代理服务器对 [illegible] 上的用户进行存取控制

用软件实现硬盘写保护卡的功能

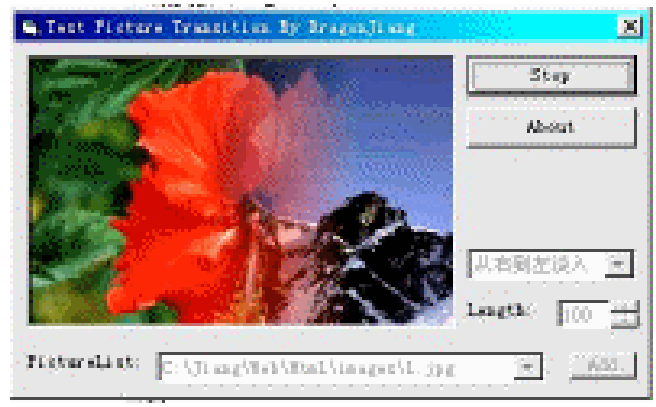
博士信箱

从中文 [illegible] 上网常见问题及其处理

本期信息窗“自在眼文件浏览器”等[illegible]篇，分别见[illegible]页。

名 誉 社 长　张　琪
社　　长　孙茹萍
副 社 长　毕研元
总　　编　王路敬
执 行 主 编　叶　永
副 主 编　杨林涛
编　　辑　程　芳　管逸群
　　张　涛　司维清
公关部主任　黄德琏
出版发行部　毕　波
编 辑 出 版　《电脑编程技巧与维护》
　　杂志社
法 律 顾 问　佟秋平 北京中律师事务所 雪
主 办 单 位　中国信息产业商会
社　　址　北京海淀区学院南路 [illegible] 号
　　吉安大厦 [illegible] 室
投 稿 信 箱
　　[illegible]
编辑部信箱
　　[illegible]
邮　　编　[illegible]
电　　话　[illegible]
传　　真　[illegible]
照　　排　《电脑编程技巧与维护》
　　杂志社电脑排版部
印　　刷　北京巨龙印刷厂
订　阅　处　全国各地邮电局
国内总发行　北京报刊发行局
邮 发 代 号　[illegible]
刊　　号　[illegible]
广告许可证　京海工商广字 [illegible] 号
全 年 定 价　[illegible] 元
每 期 定 价　[illegible] 元







希捷公司推出新版 阅[illegible]硬盘安装软件

希捷国际科技公司近日推出了其屡获奖项的 阅[illegible]硬盘安装软件的最新增强版，极大地提高了这个软件的功能和特色。新的 阅[illegible]是首个同时适用 [illegible]和 粤[illegible]硬盘的安装软件，也是随新硬盘发送的首个真 猿[illegible]位安装软件。新的 阅[illegible]具有更好的易用性，有新的安装引导用户界面、猿[illegible]彩色图形、新的交互性 与[illegible]安装手册、非常快的分区和格式化功能以及另外一些对用户非常友好的重要功能，如文件拷贝和 悦[illegible]升级。更强的编程智能性 宁[illegible]使得 阅[illegible]能更好地预测出适用不同用户的最佳安装。希捷用户可免费获得 阅[illegible]

晕[illegible]公司推出新型网络安全系统

美国互联网安全系统发展商 晕[illegible]技术公司近日在香港推出了一种专门用于互联网数据中心的新型安全系统。这个名为 晕[illegible]千兆比特）的安全系统可以支持网络防火墙和 猿[illegible]数据加密标准）、陨[illegible]网际协议安全）、宁[illegible]虚拟页码）。通过这个安全系统，粤[illegible]应用服务商）和各网络公司只需通过一个安全系统，就可以为所有个人用户提供安全管理服务，而不必象以前那样每个用户都需要一个安全设置。

第一套中文 [illegible]嵌入系统正式运行

全球著名的 [illegible]开发商 [illegible]公司近日宣布和半导体业界巨人摩托罗拉半导体部展开合作，成功地推出了在摩托罗拉 [illegible]微处理器上运行的基于 [illegible]的嵌入式系统。这个新平台于中国大饭店的因特网国际研讨会上展出。[illegible]在嵌入式中的应用已日益普及，但是在中国地区，[illegible]嵌入式系统的进展和成果还是凤毛麟角，而此次 [illegible]参考平台 [illegible]上的中文 [illegible]嵌入式系统作为国内唯一的中文 [illegible]嵌入式系统，不仅具有通常嵌入式系统占用内存小，硬盘小的优点。而且采用完全汉化的图形界面，具有高可靠性、高可伸缩性和高稳定性，极大的方便了用户。

微软计划在 宰[illegible]操作系统中引入组网技术

美国微软公司发布消息称，该公司计划在 ‘宰[illegible]’操作系统的升级版 ‘宰[illegible]’操作系统中引入局域网组网技术，从而使用户的计算机不必借助其它网络软件就可以行联网。据悉，以往的 宰[illegible]操作系统大多使用 晕[illegible]公司和 月[illegible]公司的网络互联软件，而 ‘宰[illegible]’操作系统很有可能放弃这两种软件而使用自己研制的网络客户软件。

美国推出 《大布列颠百科全书》应用软件

芝加哥 月[illegible]公司推出了 《大布列颠百科全书》应用软件。据了解，这种软件主要适用于手提电脑用户，通过它，网民可以随时上网查阅 源[illegible]万字的 《大布列颠百科全书》。这套名叫 “大布列颠旅行者” 月[illegible]的软件是 月[illegible]公司为顺应当今世界无线信息传输快速发展的趋势而做的首次尝试，目前它主要适用于 孕[illegible]型手提电脑，其它应用种类将很快投放市场。

微软将推出新型 孕[illegible]掌上电脑操作系统

微软公司宣布将在纽约和伦敦推出一种新型掌上个人电脑，同时推出的还有与之匹配的 孕[illegible]操作系统。微软以往的掌上电脑操作系统主要是 宰[illegible]，尽管该系统曾得到卡西欧、夏普和康柏等公司的支持，但始终未能撼动 孕[illegible]操作系统的统治地位。长期以来，孕[illegible]一直占据 [illegible]的市场份额，而 宰[illegible]只有 [illegible]。此番微软和康柏、惠普、卡西欧等硬件厂商再次联手推出 孕[illegible]，其目的就是向 孕[illegible]发起挑战。

陨[illegible]与 陨[illegible]开发全新安装解决方案

陨[illegible]公司宣布，它将与 陨[illegible]软件公司合作为客户提供一致的、便于使用的安装解决方案。该解决方案提供了跨平台的标准化安装流程，网络管理人员只要掌握了 ‘宰[illegible]式’安装程序，即可实现软件产品的多平台安装，从而大大提高工作效率。借助这项联合研究计划，陨[illegible]软件及运行于 粤[illegible]、粤[illegible]、[illegible]、[illegible]和 杂[illegible]平台的 陨[illegible]独立软件开发商）应用软件的最终用户将可获得与 陨[illegible]当前提供给 宰[illegible]用户一致的、便于使用的安装流程。

面向 杂[illegible]的新版 酝[illegible]集成软件面世

日前，陨[illegible]公司推出了 酝[illegible]面向 杂[illegible]的新版本。作为第二代消息处理软件，酝[illegible]可以应用企业所定义的规则和智能化路由自动选择信息，并将其发送给应用软件、数据库以及用户。酝[illegible]提供了定义处理关健数据或业务事件的图形工具，而且可以动态地控制和发送消息，并使其与企业的数据集成起来，最终分发给商务应用软件。它不但提供了对 杂[illegible]和 韩[illegible]等行业标准的支持，而且拥有了规则与格式化程序引擎，使用户可以全天候地部署动态应用。

瑞典一公司开发 酝[illegible]挑战 陨[illegible]

瑞典 酝[illegible]公司推出了一种名为 酝[illegible]的软件，它使互联网用户可以在同一界面下通过多个信息传递途径进行交流和信息传递。目前该软件可以支持美国在线公司的 陨[illegible]和及时信使 粤[illegible]等信息传递方式。酝[illegible]公司在其服务网站上刊登广告称，‘酝[illegible]软件使用一种极其先进的连接技术，使其成为及时信息传递服务的最佳选择，用户将可以通过

该软件与其他任何信息传输途径相兼容，包括 [illegible] 和 [illegible]。”但到目前为止，美国在线却拒绝就该软件发表评论。

[illegible] 公司推出多功能新型浏览器

美国 [illegible] 公司推出了一种新型网络交互浏览器，使用这种浏览器，用户可以在同一个页面聊天、留言或浏览网站。当访问者点击 [illegible] 的主页时，屏幕上将出现一个新窗口，在这个窗口中，访问者可以与其它正在观看同一网页的人进行交谈。点击 [illegible] 的网站，也将会出现一个新窗口，在这里，网站可以与访问者交换信息。

华工研发出“能听会说”语音网络套餐

湖北省华中理工大学自主研究开发出“能听会说”的“语音网络套餐”，给使用中文上网冲浪的网民带来了便利。华中理工大学计算机学院在与 [illegible] 等公司合作的基础上，由陈刚等人自主研究开发出的这套名为“语音网络套餐”的软件包括三个部分：语言浏览器、语音邮件和语音通讯簿。

诺方与 [illegible] 合作实现电子讯息安全服务

日前，诺方公司宣布与 [illegible] 结成战略合作伙伴关系，帮助其实现电子讯息安全服务。[illegible] 在由第三方实现的一体化的电子邮件服务领域中居于世界领先地位。此次合作将使各类企业和机构能够利用 [illegible] 的基于 [illegible] 的电子邮件服务在任何地方轻松、经济的收发保密和安全的电子邮件。通过此次双方的合作，用户可以利用一个易用的，基于 [illegible] 的解决方案，以经过验证的身份在企业内部和企业之间实现电子讯息的收发，并同时保证个人的隐私和信息的完整与真实。

[illegible] 推出新系统 让你动嘴就能上网

[illegible] 公司正式发布最新 [illegible] 世纪版中文语音识别系统，据称这是该系统第五次更新发布、也是具有里程碑式的最新版本，它使人机交互变得更轻松、更容易，也更有人情味。据介绍，最新版本语音系统在功能、识别率、易用性等方面都较以往有大幅提高，特别加强了网络应用比重。除了网上浏览对话以外，用户还可以口述文档和电子邮件以及进行文档格式编排等。普通百姓也照样“只动口不动手”地操作电子设备，网上冲浪照样轻松自如。

莲花公司将推出新型 [illegible] 知识管理软件

莲花公司将于今年内推出一种名为 [illegible] 的新型知识管理软件，安装这种软件之后，各公司将可以完全控制公司雇员所掌握的情况。莲花公司称，随着近年来劳动力市场发生的巨大变化，公司很难将某个雇员留到 [illegible] 年以上，但却有一个方法将雇员在公司内所积累的工作经验留下来。在雇员离开公司之后，[illegible] 知识管理软件可以为公司重新获得该名雇员在任职期间所掌握的情况。它可以帮助公司进入雇员的电子邮件、语音邮件以及其它工作资料。

[illegible] 新版数据库全面集成电子商务

[illegible] 公司发布了业界第一个具备全面集成能力的电子商务数据库——[illegible] 通用数据库 [illegible] 版。新产品为 [illegible] 电子交易提供了强大支持。它不但可以使从事企业对企业 [illegible] 电子交易的网站和企业敏锐地洞察客户需求、缩短应用软件的部署时间，而且使用户具备高速因特网查询能力，其速度将比传统关系数据库搜索引擎快 [illegible] 倍。作为业内唯一集电子商务、商业智能与内容管理功能于一身的数据库，[illegible] 版为用户提供了更高的性能和更广泛的可用性。

美国发明小如尘埃的“智能电脑”

最近，一种长约 [illegible] 毫米、由多个微型电子机械系统组成的微型电脑，由美国加州大学伯克利分校的科学家发明成功，并在西雅图举行的移动通讯展览会上亮相。这种“智能尘埃”由感应器、激光器和收发讯器等几种精密装置构成。作为一种电脑监察装置，它可用来执行间谍任务，同时还能参与观测天气和侦测化学毒气，在军事上也有广阔的用途。科学家通过实验发现，可以在 [illegible] 公里以外对“智能尘埃”进行遥控操作，收集它们传回的资料信息，其传输速率为数千字节 [illegible] 每秒。

新加坡推出虚拟房地产软件

美国 [illegible] 地产公司发明的虚拟房地产软件，可足不出户，在网上立体参观所欲购买房屋的每一角落，并可更改房屋设计和家具，随心所欲地创造一个称心如意的居住环境。只要启动这个软件，电脑屏幕上就会出现住屋大门的景观，再用鼠标沿住屋方向依次轻点，就可分别参观客厅、卧室、厨房、卫生间等。

[illegible] 推出 [illegible] 超高速二极管

国际整流器公司 [illegible] 简称 [illegible] 推出全新 [illegible] 外延二极管系列，其反向恢复时间仅为 [illegible] 这些超高速恢复二极管有单独及中心抽头双重配置，适用于 [illegible] 开关电源 [illegible] 的输出级。此类应用是目前电源市场中发展最快的领域之一。欲查询详细资料，或索取有关数据表，可浏览 [illegible] 公司的网页 [illegible]。

三洋电机开发出消耗电力小至 [illegible] 的 [illegible] 码 [illegible]‘[illegible]’

芯片完全按接线逻辑来实现解码电路的，所以可实现 [illegible] 动作频率并降低了耗电。目前用于便携式音乐唱机的音频编码方式除了 [illegible] 以外，还有 [illegible] 等多种规格。为此，三洋电机公司还着手开发 [illegible] 解码 [illegible]。据说该 [illegible] 可以同时解码 [illegible] 使用的 [illegible] 两种方式。该产品适用于便携式音乐唱机。

需要更多请访问 www.ertongbook.com

使用 阅读器创建程序组和程序项

吴 蕾 范丽华

一个商品化的软件都提供安装程序，用户只需要运行程序 安装好 安装 或 阅读器 确认版权信息、输入产品序列号及选择不同的安装方式，即可按提示完成软件的安装。

一、阅读器接口

为了方便用户使用，还必须创建程序文件夹和快捷键。宰蛮 是利用 马列斯 配莱斯 的动态数据交换 阅读器 接口来创建程序文件夹和快捷键的。为了帮助读者理解 阅读器 的机制，我们首先要简单的介绍一下 阅读器 的概念。

动态数据交换是 宰蛮 应用程序在运行时数据交换的一种技术，它是由一组 阅读器 协议构成，并由 阅读器 动态库所支持。在一次 阅读器 动态交互过程中，进行交互的一方称为客户方，另一方称为服务器方 而且一次 阅读器 交互总是从客户端发起，由服务器提供各种数据服务。每个服务器有唯一的一个名字，称之为服务名，由一个字符串表示，客户端通过此服务名字符串与服务器建立连接。服务器提供的数据由主题名划分成不同的数据集，并由项名唯一的指定某个数据集中的某个数据 （主题名和项名都由字符串表示）。客户程序使用服务器名和主题名与服务器建立会话连接，在连接建立之后使用项名请求服务器提供数据。在 阅读器 中一次数据或服务请求被称为事务，其主要包括以下事务 : 菲斯阿姆特雷尔 菲斯阿姆特雷尔 菲斯阿姆特雷尔 菲斯阿姆特雷尔 菲斯阿姆特雷尔 阅读器 接口主要包含以下接口函数：

初始化

阅读器 程序在进行 阅读器 交互之前，必须调用函数 阅读器 进行初始化。在初始化函数中指定 阅读器 回调函数，以便使 阅读器 能够将 阅读器 事务传递给程序。函数调用成功之后返回 阅读器 在 阅读器 程序结束运行之前，必须调用函数 阅读器 释放与应用程序有关的所有 阅读器 库资源。

字符串管理

为建立会话和执行事务，阅读器 程序之间需要交换字符串。函数 阅读器 就是进行字符串管理的函数。这个函数创建一个句柄，在程序中使用这个句柄代表字符串。阅读器 客户和服务器应用程序可以把字符串句柄做为参数传递给其它 阅读器 函数。使用完字符串句柄后，应调用函数 阅读器 来释放字符串句柄。

数据管理

在程序之间交换的数据必须存储在 阅读器 库提供的函数分配和管理的内存对象中。函数 阅读器 创建一个全局内存对象。使用完此对象应调用 阅读器 函数删除此全局内存对象。

会话管理

客户和服务器之间的会话由客户发起。当一个会话被成功的发起之后，客户和参与服务器都会得到一个会话句柄，双方都使用此句柄发送事务和管理这个会话。客户通过调用 阅读器 函数与一个由调用参数指定的服务器建立会话。在任何时候，客户或服务器都可以调用 阅读器 函数终止会话。

事务管理

在客户和服务器建立会话连接之后，客户应用程序可以调用函数 阅读器 在客户和服务器程序之间启动数据事务处理，通过发送一个事务来从服务器中获取数据或服务

回调函数

参与 阅读器 会话的应用程序必须定义一个回调函数，以便使 阅读器 能够将 阅读器 事务传递给该函数。

错误处理

当 阅读器 函数失败时，程序可以调用函数 阅读器 来确定产生错误的原因。

二、菲斯阿姆特雷尔 动态数据接口

马列斯 配莱斯 有一个 阅读器 命令字符串接口。此接口允许应用程序创立、删除和重新安装程序组，给程序组添加或删除、替换项目。为了完成上述目的，马列斯 配莱斯 提供了以下命令：悦耳 阅读器 贵宾 阅读器 杂乱 阅读器 粤语 阅读器 阅读器 阅读器 每条命令必须用方括号括起来，参数放在圆括号里，并用逗号分开，并且多个命令可以连续使用。

为了执行这些命令应用程序必须先和 马列斯 配莱斯 建立 阅读器 会话。马列斯 配莱斯 作为 阅读器 服务器，应用程序作为 阅读器 客户端。服务器的服务名和主题名都是 马列斯。应用程序和 马列斯 配莱斯 建立会话之后就可以向其发送命令字符串。

以下先对经常要使用的部分 马列斯 配莱斯 阅读器 命令字符串做一些解释：

贵宾 阅读器 贵宾

创建一个新程序组或激活一个现存的程序组。其格式为：

CreareGroup(组名[, 组标识])

圆桌 杂乱 阅读器 贵宾

使现有的组的窗口最大化、最小化或复原。其格式为：

ShowGroup(组名, 窗口特征标识[, 组标识])

猿猴 粤语 阅读器 贵宾

删除现有的组。其格式为：

DeleteGroup(组名[, 组标识])

源程序如下：

在已经存在的组中创建一个程序项。其格式为：

AddItem(命令[,标题[,图标文件名[,图标在图标文件的索引[,组窗口中的X坐标[,组窗口中的Y坐标[,默认目录名[,热键[,应用程序第一次显示时是否最小化标志]]]]]]])

三、应用实例

我们用下面的例子说明利用动态数据接口创建程序组和程序项的步骤和技巧。在此例中我们将创建一个程序项。

首先利用应用程序向导创建基于对话框得工程文件生成如下类和实现文件：

类名	基类	头文件	实现文件
对话框类	应用程序类	对话框头文件	对话框实现文件
对话框类	对话框类	对话框头文件	对话框实现文件
对话框类	对话框类	对话框头文件	对话框实现文件

接着修改对话框，加入如下控件：

控件名	标题
对话框按钮	建立图标
对话框按钮	删除图标
对话框按钮	测试类型
对话框	确定
对话框	取消

然后利用应用程序向导向对话框类中添加处理函数和变量。

为对话框按钮添加单击事件的处理函数如下：

为对话框按钮添加单击事件的处理函数如下：

为对话框按钮添加单击事件的处理函数如下：

添加变量如下：

另外，在对话框类中添加私有变量如下，类型为全局函数如下：

源程序实现以上命令处理函数和全局处理函数。

```
HDDEDATA CALLBACK DdeCallback
    ( UINT uType,  // transaction type
      UINT uFmt,   // clipboard data format
      HCONV hconv,  // handle to the conversation
      HSZ hsz1,   // handle to a string
      HSZ hsz2,   // handle to a string
      HDDEDATA hdata, // handle to a global memory object
      DWORD dwData1,  // transaction-specific data
      DWORD dwData2  )
    {
        return 0;
    }
void CNewDdeDlg::OnCreate()
{
 // TODO:  Add your control notification handler code here
 m_Index = 1; //修改变量的值以指示是添加还是删除程序项
  }
void CNewDdeDlg::OnDelete()
{
 // TODO:  Add your control notification handler code here
 m_Index = 2; //修改变量的值以指示是添加还是删除程序项
}
void CNewDdeDlg::OnOk()
{
 // TODO:  Add your control notification handler code here
 DWORD  idInst=0;
    HCONV     hConv=0;
    HDDEDATA  hData;
    HSZ       hszService=0, hszTopic=0;
    FARPROC   lpCallBack;
 if(m_Index==1)//创建程序组和程序项
 {
  lpCallBack=MakeProcInstance((FARPROC)DdeCallback,
        AfxGetInstanceHandle()); //取得指向回调函数的指针
  if(lpCallBack==NULL)
       return;
  if(DdeInitialize(&idInst, (PFNCALLBACK) lpCallBack, AP-
PCMD_CLIENTONLY, 0L)!=
       DMLERR_NO_ERROR)//DDE 程序初始化,指定回调函数
            return;
 hszService=DdeCreateStringHandle(idInst, "PROGMAN", CP_
WINANSI);
 hszTopic=DdeCreateStringHandle(idInst, "PROGMAN", CP_
WINANSI);
 if(hszService==NULL||hszTopic==NULL)
       return;
 hConv=DdeConnect(idInst, hszService, hszTopic, NULL);
 //与服务器建立连接
 if(hConv==NULL)
       return;
 CString cmmd;
 LPTSTR  temp = new char[50];
 GetCurrentDirectory(50, temp);
 cmmd=temp;
 cmmd+="\\NewDde.exe";
 char cmd[100]="[CreateGroup(\"DDE 演示程序\")]
[AddItem(\"";
 strcat(cmd, cmmd);
 strcat(cmd, "\", \"DDE 演示\")]";
 //生成 Program Manager DDE 命令串
 hData=DdeCreateDataHandle(idInst, (LPBYTE) cmd, sizeof
(cmd), 0, 0, CF_TEXT, 0);
 if(DdeClientTransaction((LPBYTE)hData, -1, hConv, 0, 0,
                                        //发送 DDE 命令
 XTYP_EXECUTE, TIMEOUT_ASYNC, NULL)==0)
                AfxMessageBox("Error");
 }
 if(m_Index==2)
 {
   lpCallBack=MakeProcInstance((FARPROC)DdeCallback,
  AfxGetInstanceHandle()); //取得指向回调函数的指针
  if(lpCallBack==NULL)
       return;
  if(DdeInitialize(&idInst, (PFNCALLBACK) lpCallBack, AP-
PCMD_CLIENTONLY, 0L)!=DMLERR_NO_ERROR)
```

VB 中与通讯口交互的实现方法

冯益明　洪玲霞　刘洪霞

摘　要　本文介绍一种在 VB 中与通讯口交互信息的一种方法与技巧，以解决在程序开发过程中需要与计算机外围设备进行交互的问题。

一、引言

在计算机程序设计过程中，经常会遇到与外围设备进行交互的问题，如读鼠标、数字化仪等外围设备的串口信息以及读打印机等并口信息，应用程序通过对计算机串、并口的读写操作，满足用户需求。本文仅以读写调制解调器信息为例进行论述。

二、应用程序的编写

在 VB 集成环境下新建一标准 EXE 工程，工程名为 Project1，工程文件名为 Project1.vbp，窗体名为 Form1，窗体文件名为 Form1.frm。窗体属性 Caption 为“MSComm 电话拨号器”。在工程菜单部件中选择添加 Microsoft Comm Control 6.0，将 MSComm 控件添加到窗体中添加如图 1 所示的各种控件。控件 Command1 的 Caption 属性为“拨号”，控件 Command2 的 Caption 属性为“取消”，控件 Command3 的 Caption 属性为“退出”，控件 Label1 的 Caption 属性为“要拨号码，请击‘拨号’按钮”，将控件 MSComm1 的 InputLen 属性设为 1，控件 MSComm1 的属性值采用控件默认属性。

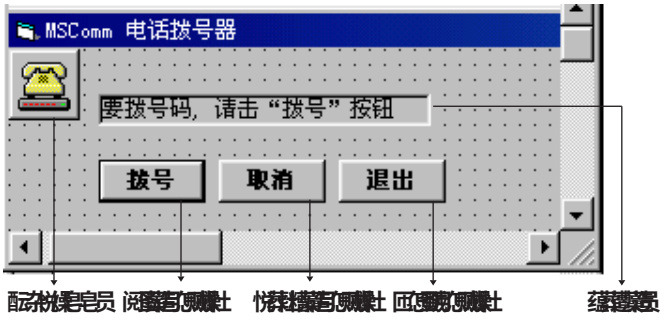


图 1

程序实现代码如下：

```
‘ Comm.FRM
’　演示如何用调制解调器拨电话号码.
’　要求电话和调制解调器用同一根电话线相连
DefInt A－Z
’ 当用户选择取消时设置此标志.
Dim CancelFlag
’调用窗体 Commr 的事件
Private Sub Form_Load()
    ’ 告知 MSComm1 读取输入缓冲区中的全部内容.
    MSComm1.InputLen = 0
End Sub
‘‘取消’按钮单击事件
Private Sub CancelButton_Click()
    ’ CancelFlag 用来告知退出拨号过程.
    CancelFlag = True
    CancelButton.Enabled = False
End Sub
’‘退出’按钮单击事件
Private Sub QuitButton_Click()
    End
End Sub
‘‘拨号’按钮单击事件
Private Sub DialButton_Click()
    Dim Number$, Temp$
    DialButton.Enabled = False
    QuitButton.Enabled = False
    CancelButton.Enabled = True
    ’ 取得要拨的电话号码.
    Number$ = InputBox$(″请输入电话号码:″, Number$)
        If Number$ = ″″ Then Exit Sub
    Temp$ = label1
```

```
//DDE 程序初始化,指定回调函数
                return;
hszService = DdeCreateStringHandle(idInst, ″PROGMAN″, CP_
WINANSI);
    hszTopic = DdeCreateStringHandle(idInst, ″ PROGMAN″,
CP_WINANSI);
    if(hszService = =NULL||hszTopic = =NULL)
        return;
    hConv = DdeConnect(idInst, hszService, hszTopic, NULL);
//与服务器建立连接
    if(hConv = =NULL)
        return;
    char cmd[100] = ″ [DeleteGroup(\″ DDE 演示程序\″)]
[DeleteItem(\″DDE 演示\″)]″;
    hData = DdeCreateDataHandle(idInst, (LPBYTE) cmd, sizeof
(cmd), 0, 0, CF_TEXT, 0);
if(DdeClientTransaction((LPBYTE) hData, －1, hConv, 0, 0,
XTYP_EXECUTE, TIMEOUT_ASYNC, NULL) = =0)
        AfxMessageBox(″Error″);
    }
}
```

参考文献

陈坚，陈涛 .利用 VB 创建可编制 宏的应用程序 .西安电子科技大学出版社 .

收稿日期：（年 月 日）

```
    Label1 = "正在拨号码 - " + Number$
    ' 拨选定的电话号码.
    Dial Number$
    DialButton.Enabled = True
    QuitButton.Enabled = True
    CancelButton.Enabled = False
    Label1 = Temp$
End Sub
'定义一拔号子过程
Private Sub Dial(Number$)
    Dim DialString$, FromModem$, dummy
    ' AT 是贺氏兼容 ATTENTION 命令并且要求发送命令到调制解调器.
    ' DT "音频拨号"; DP = "脉冲拨号"
    ' Numbers$ 为要拨的电话号码.
    ' 分号告知调制解调器拨号后返回命令
    ' 一个回车, vbCr, 当发送命令到调制解调器时被要求.
    DialString$ = "ATDT" + Number$ + ";" + vbCr
    ' 假设"调制解调器"端口设置为 COM2
    MSComm1.CommPort = 2
    MSComm1.Settings = "9600, N, 8, 1"
    ' 打开通信端口
    On Error Resume Next
    MSComm1.PortOpen = True
    If Err Then
        MsgBox " COM2: 不可用. 更改 CommPort 属性使其使用其它串口"
        Exit Sub
    End If
    ' 清空输入缓冲区.
    MSComm1.InBufferCount = 0
    ' 拨电话号码.
    MSComm1.Output = DialString$
    ' 等待调制解调器返回信息
    Do
        dummy = DoEvents()
        ' 如果缓冲区中有数据, 则把它读出来.
        If MSComm1.InBufferCount Then
            FromModem$ = FromModem$ + MSComm1.Input
            If InStr(FromModem$, "OK") Then
                ' 提醒用户摘电话.
                Beep
                MsgBox "请您摘电话然后按 '回车 '键或单击 '确定 ""
                Exit Do
            End If
        End If
        ' 用户是否选择 '取消 "?
        If CancelFlag Then
            CancelFlag = False
            Exit Do
        End If
    Loop
    ' 断开与调制解调器的连接.
    MSComm1.Output = "ATH" + vbCr
    ' 关闭端口.
    MSComm1.PortOpen = False
End Sub
```

三、结束语

本文较为详细的提供了一种与计算机外围设备原调制解调器进行信息交互的例子，为程序员开发有关计算机与通讯口交互方面的程序提供了一个好的例子。

收稿日期：□□□□年 □月 □□日）



需要全部请访问 www.ertongbook.com

创建 宰蜜蹴糟择怨辮辘愿快捷方式技巧

洪志全

随着计算机硬件性能的提高，各种应用软件的开发已经完全从 配劝原瑞柒平台过渡到 宰蜜蹴糟择平台。在 宰蜜蹴糟择怨辮辘愿平台中，当一个应用软件开发完成后，安装程序制作是每一个应用软件所必须的。当然，可以选择商品化的安装程序制作工具，但一般为了表现自己软件的特点往往自己设计相应的安装程序。宰蜜蹴糟择怨辮辘愿桌面快捷方式可简化了用户启动程序、访问文件操作，所以应用软件在安装的最后阶段都会在开始菜单或者桌面上建立应用程序的快捷启动方式。在安装程序或在应用程序中如何动态地添加、删除应用程序在 宰蜜蹴糟择怨辮辘愿桌面和开始菜单中的快捷启动方式是非常重要的。本文将介绍如何在 灾蜜蹴造月择蜜糟缘园中动态添加 辅删除 宰蜜蹴糟择的快捷启动方式的方法。

一、创建快捷文件夹

宰蜜蹴糟择怨辮辘愿快捷文件夹也就是 宰蜜蹴糟择怨辮辘愿目录中 怨辮删晒砸漆怎和 怨辮删晒砸漆怎赞列辮删晒彩择目录下的应用程序子目录，这一般可通过 灾蜜蹴造月择蜜糟的 悦澡删命令和 配澡删命令来实现。一般来说，创建快捷文件夹首先需获取 宰蜜蹴糟择怨辮辘愿系统目录，使用 宰蜜蹴糟择怨辮辘愿的 粤孕陷函数 则辅宰蜜蹴糟择删增删来实现，创建快捷文件夹示例程序如下：

```
Private Declare Function GetWindowsDirectory Lib ″kernel32″
(ByVal lpBuffer As String, ByVal nSize As Long) As Long
Private Sub MakeProgramGroup()
  Dim WinNo As Integer
  Dim WinDirectoryString As String, ShortCutPath As String
  WinDirectoryString = Space(255)
   WinNo = GetWindowsDirectory(WinDirectoryString, Len
(WinDirectoryString))
  WinDirectoryString = Trim(WinDirectoryString)
  ShortCutString = ″Hong Zhiquan″
  ShortCutPath = Left(WinDirectoryString, Len(WinDirectory-
String) – 1) + ″\Start Menu\Programs″
  ChDir ShortCutPath
  MkDir ShortCutString
End Sub
```

调用上述程序，可在 ‘开始→程序 ”中创建 匀操早在蹿蹴社的快捷文件夹。若将 怨辮删晒砸漆怎改写为 怨辮删晒彩择越 缘蜜蹴造月择蜜糟宰蜜蹴糟择删增删晒砸漆怎雪原 员雪垣 薰赞删晒配澡怎薰 便可在 ‘开始 ”菜单中创建 匀操早在蹿蹴社快捷文件夹。用类似方法，使用 灾月的 砸澡删命令也可删除已创建的快捷文件夹。

二、创建程序快捷

在 宰蜜蹴糟择怨辮辘愿中，一个应用程序的快捷方式实际上是一个包含该应用程序启动信息的连接文件 （ 援蹿蹴 ），创建应用程序快捷也就是创建该应用程序的连接文件。连接文件存放于不同的文件夹，就对应不同的快捷方式，如连接文件存放于 揍宰蜜蹴糟择蹿蹴删晒砸漆怎薰目录，对应为 宰蜜蹴糟择桌面快捷方式。在 灾蜜蹴造月择蜜糟中创建 宰蜜蹴糟择怨辮辘愿的应用程序快捷，可用 灾月辅提供的动态链接库 灾蜜蹴删晒砸漆怎进行。灾蜜蹴删晒砸漆怎动态链接库提供了 宰删蹴蹿怎漆怎蜜蹴操 宰删蹴蹿怎漆怎蜜蹴桌函数，用于创建程序快捷和删除程序快捷。这两个函数的调用形式如下：

```
Private Declare Function fCreateShellLink Lib ″ Vb5stkit. dll″
(ByVal lpstrFolderName As String, ByVal lpstrLinkName As
String, ByVal lpstrLinkPath As String, ByVal lpstrLinkArgu-
ments As String) As Long
Private Declare Function fRemoveShellLink Lib ″ Vb5stkit. dll″
(ByVal lpstrFolderName As String, ByVal lpstrLinkName As
String) As Long
```

其中 遗删蹴蹿怎漆怎蜜蹴操笔藻为保存快捷的文件夹，默认为薰宰蜜蹴糟择怨辮辘愿删晒砸漆怎赞列辮删晒彩择目录；遗删蹴蹿怎漆怎蜜蹴操笔藻快捷程序描述，遗删蹴蹿怎漆怎蜜蹴操笔藻为快捷所对应的应用程序。

员援在桌面创建快捷

在 灾月下新建 “工程 ”，然后在 云删蹴员的 ‘通用 ”的 ‘声明 ”中输入上面的两个 粤孕陷函数声明，然后输入下面子程序：

```
Private Sub MakeDeskTopShortCut(AppProgName As String,
AppProg As String)
  Dim lresult As Long
  lresult = fCreateShellLink(″ . . \ . . \ Desktop″, AppProg-
Name, AppProgram, ″″)
End Sub
```

调用 配澡蹿蹴操怎蹿蹴删晒砸漆怎蜜蹴操程序即可在 宰蜜蹴糟择怨辮辘愿桌面创建应用程序快捷。

圆援在开始菜单创建快捷

用在桌面创建快捷类似的方法，将保存快捷的文件夹薰赞揍删晒砸漆怎薰替换为薰赞揍赞揍删晒砸漆怎薰更可在开始菜单创建快捷，如：遗删蹴蹿越 宰删蹴蹿怎漆怎蜜蹴操笔藻薰赞揍赞揍删晒砸漆怎薰 粤孕陷函数声明笔藻 粤孕陷函数声明薰 薰薰； 若替换为薰赞揍赞揍删晒砸漆怎赞列辮删晒彩择薰更可在 ‘开始→程序 ”中添加快捷启动方式；若已经创建快捷文件夹，将保存快捷的文件夹替换为薰赞揍赞揍删晒配澡怎赞列辮删晒彩择薰与快捷文件夹名，即可在 ‘开始→程序→快捷文件夹名 ”中创建应用程序快捷启动方式。

在 Windows下对文件与目录进行编程

杨 鹰

近来，随着 Windows系统的不断普及，使用 Windows系统的单位和个人也越来越多，特别是 Windows的推出，更是风靡全球。基于 Windows系统的编程也愈显重要。本文以一个简单的程序为例，说明怎样在 Windows系统下对文件和目录进行编程。该例是一个类似于 Windows系统命令 dir的程序，用于列出指定的目录中所有文件和子目录并计算出指定目录所用的磁盘空间。

一、 Windows文件系统简介

文件是通过操作系统来管理的。文件的结构以及命名、存取、使用、保护和实现方法都是 Windows系统中的重要内容。总体上，Windows系统中处理文件的那部分称为文件系统。同 Unix类似，Windows中的文件系统被组织成树状结构。

Windows中的文件和目录通过一个 32比特的保护码来进行保护。保护码分成三个 9比特的域，分别对应着文件主、同组用户和其他用户。每个域有一位标识读权限，一位标识写权限，一位标识执行权限，这三位即 rwx位。

在 Windows文件系统中通常包含如下文件类型：

普通文件：一个文件包含所有用户放在其中的信息。它可以被认为是一个字节序列。它与 Unix及 Macintosh系统不同，对于每个文件都具有一定的读写权限。

目录：目录提供了文件名称与文件自身之间的映射，由此使得文件系统上的一个结构被作为一个整体对待。一个目录可包含文件，也可以包含子目录，这些子目录同样可以包含更多的文件和子目录。一个目录在被读的时候，它的行为完全像一个普通文件，但它不能被没有权限的用户程序访问。

特殊文件：特殊文件是 Windows文件系统中最有特色的地方之一。每个 I/O设备（磁盘驱动器、终端等）都与一个这样的文件有关。对特殊文件的操作和普通文件一样，但它引发了对相关设备的操作。特殊设备文件的实体保存在目录 /dev中。

符号链接文件：一个符号链接在行为上就像指向另一个文件的指针，这有点像 C语言中的指针。在文件系统中，实现这一点是建立一个带有链接名称的文件，该链接指向文件的路径名。

i-node标识号，i-node列表和 i-node节点：一个目录是由一系列结构组成的，每个结构包含有一个文件名和一个指向文件自身的指针，该指针是一个整数，称为文件的 i-node标识号。当文件被访问时，它的 i-node标识号用来作为索引打开一个系统表（i-node列表），系统表中存放着文件（i-node节点）的实体。i-node节点中包含了如下对文件的描述信息：

- 文件自身的用户和用户组 ID
- 文件的保护码
- 文件内容所在的物理磁盘地址
- 文件的大小
- 最后一次 i-node节点改变的时间，最后一次使用和最后一次修改的时间
- 连接该文件的次数，即它出现在其它目录中的次数
- 一个指明文件类型的标记（目录、普通文件或特殊文件）

二、 有关的系统调用介绍

stat()系统调用：这是一个非常有用的系统调用，用来获取存储在一个 i-node节点上的信息。它接受两个参数：一个字符串指针，指向说明一个文件的路径名；另一个是指向 stat结构的指针，在这个结构中，存放着有关这个文件的信息。stat结构包括以下成员：

st_mode: 这个字段包含文件类型和它所具有的访问权限。

st_ino: 这个字段在一个给定的文件系统中唯一的标识了这个文件。

st_dev: 这个字段唯一地标识了包括这个文件的文件系统。

st_rdev: 如果 i-node节点是一个特殊设备文件，则这个字段标识设备的类型。

st_nlink: 文件链接的个数。

st_uid: 文件属主的用户 ID。

st_gid: 文件属主的组 ID。

st_size: 以字节为单位的文件长度。

st_atime: 文件最后被访问的时间。

st_mtime: 文件内容最后被修改的时间。

st_ctime: 文件 i-node最后被修改的时间。

lstat()系统调用：这个系统调用与 stat()系统调用非常相似，唯一的区别是如果文件是一个符号链接，lstat()将返回关于这个符号链接的信息，而不是该符号链接引用的文件。

chown()系统调用：这个系统调用允许改变一个文件的用户 ID和组 ID。

chmod()系统调用：这个系统调用允许改变一个文件的访问权限和其他标志位。

utime()系统调用：这个系统调用允许改变一个 i-node节点中的 st_atime和 st_mtime字段。

opendir()系统调用：这个系统调用打开一个目录，为后续的读操作做好准备。

readdir()系统调用：这个系统调用从一个目录中读出一个表项。

closedir()系统调用：这个系统调用关闭一个目录。

mkdir()系统调用：这个系统调用创建一个新目录。

rmdir()系统调用：这个系统调用删除一个空目录。

chdir()系统调用：这个系统调用改变当前工作目录。

link()系统调用：这个系统调用创建一个指向一个文件的链接。

unlink()系统调用：这个系统调用删除一个目录项，并减少 i-node节点中的链接数。如果这是文件的最后一个链接，则该文件也从文件系统中删除。

symlink()系统调用：这个系统调用创建一个符号链接。

readlink()系统调用：这个系统调用读取一个符号链接的值。

mknod()系统调用：这个系统调用创建一个特殊设备文件。

rename()系统调用：这个系统调用改变一个文件或目录的名称，可以在同一个目录或不同目录间移动。

mount()系统调用：这个系统调用允许将一个新的文件系统挂接到现有的文件系统中。

umount()系统调用：这个系统调用将一个文件系统从系统中卸载。

sync()系统调用：这个系统调用将内存缓冲区中的所有修改过的块立即写到磁盘上。

三、 Windows系统调用的实现

Windows系统调用的实现是通过 Windows内核中的一组过程来实现的。这些过程通常称为 Windows API。Windows API是 Windows系统的一个重要组成部分，它为 Windows应用程序提供了访问 Windows内核的接口。Windows API包括了大量的函数，这些函数可以分为以下几类：

进程管理：这类函数用于创建、终止、同步和管理进程。

线程管理：这类函数用于创建、终止、同步和管理线程。

内存管理：这类函数用于分配、释放和管理内存。

文件管理：这类函数用于创建、打开、读写、关闭和管理文件。

设备管理：这类函数用于访问和管理设备。

图形用户界面：这类函数用于创建和管理图形用户界面。

网络管理：这类函数用于网络通信和管理。

安全管理：这类函数用于安全和权限管理。

Windows API函数的调用方式与普通函数调用类似，但是它们是通过 Windows内核来执行的。Windows内核负责将这些函数调用转换为对硬件的操作。

Windows API函数的参数和返回值通常是指针或句柄。句柄是一个整数，用于标识一个对象，例如文件、进程、线程等。指针是一个内存地址，用于访问内存中的数据。

Windows API函数的错误处理通常是通过返回值来实现的。如果函数调用失败，则返回值通常是一个错误代码。可以使用 GetLastError()函数来获取错误代码。

Windows API函数的文档非常详细，可以在 Microsoft Developer Network (MSDN)上找到。MSDN是 Microsoft提供的一个在线文档库，包含了 Windows API、.NET Framework、Visual Studio等的文档。

四、 Windows编程实例

下面是一个简单的 Windows编程实例，该程序创建一个窗口，并在窗口中显示一个消息框。

```
#include <windows.h>

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam);

int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hPrevInstance, LPSTR lpCmdLine, int nCmdShow)
{
    const char CLASS_NAME[] = "Sample Window Class";

    WNDCLASS wc = {};

    wc.lpfnWndProc = WindowProc;
    wc.hInstance = hInstance;
    wc.lpszClassName = CLASS_NAME;

    RegisterClass(&wc);

    HWND hwnd = CreateWindowEx(
        0,
        CLASS_NAME,
        "Learn to Program Windows",
        WS_OVERLAPPEDWINDOW,
        CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT, CW_USEDEFAULT,
        NULL,
        NULL,
        hInstance,
        NULL
        );

    if (hwnd == NULL)
    {
        return 0;
    }

    ShowWindow(hwnd, nCmdShow);

    MSG msg = {};
    while (GetMessage(&msg, NULL, 0, 0))
    {
        TranslateMessage(&msg);
        DispatchMessage(&msg);
    }

    return 0;
}

LRESULT CALLBACK WindowProc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam)
{
    switch (uMsg)
    {
    case WM_DESTROY:
        PostQuitMessage(0);
        return 0;

    case WM_PAINT:
        {
            PAINTSTRUCT ps;
            HDC hdc = BeginPaint(hwnd, &ps);

            FillRect(hdc, &ps.rcPaint, (HBRUSH) (COLOR_WINDOW+1));

            EndPaint(hwnd, &ps);
        }
        return 0;

    case WM_LBUTTONDOWN:
        MessageBox(hwnd, "Hello, Windows!", "Hello", MB_OK);
        return 0;
    }

    return DefWindowProc(hwnd, uMsg, wParam, lParam);
}
```

该程序首先注册一个窗口类，然后创建一个窗口。窗口创建后，程序进入消息循环，等待用户输入。当用户在窗口中单击鼠标左键时，程序显示一个消息框，其中包含文本 "Hello, Windows!"。

五、 Windows编程工具

Windows编程可以使用多种工具，例如 Visual Studio、Code::Blocks、Dev-C++等。Visual Studio是 Microsoft提供的一个集成开发环境，包含了编辑器、编译器、调试器等。Code::Blocks和 Dev-C++是开源的集成开发环境，也包含了编辑器、编译器、调试器等。

Visual Studio提供了强大的调试功能，可以单步执行程序、设置断点、查看变量的值等。Code::Blocks和 Dev-C++也提供了调试功能，但是不如 Visual Studio强大。

Visual Studio提供了资源编辑器，可以方便地创建和编辑对话框、菜单、图标等资源。Code::Blocks和 Dev-C++也提供了资源编辑器，但是不如 Visual Studio强大。

Visual Studio提供了 MFC（Microsoft Foundation Classes）类库，可以方便地创建 Windows应用程序。MFC是一个面向对象的类库，封装了 Windows API，使得编程更加容易。Code::Blocks和 Dev-C++也可以使用 MFC，但是需要手动配置。

六、 Windows编程资源

Windows编程的资源非常丰富，可以在 Microsoft Developer Network (MSDN)上找到大量的文档、示例代码、教程等。MSDN是 Microsoft提供的一个在线文档库，包含了 Windows API、.NET Framework、Visual Studio等的文档。

CodeProject是一个非常流行的编程网站，提供了大量的 Windows编程文章和示例代码。

Stack Overflow是一个问答网站，可以在上面提问和回答 Windows编程问题。

GitHub是一个代码托管平台，可以在上面找到大量的 Windows编程开源项目。

七、 Windows编程的未来

Windows编程的未来是光明的。随着 Windows系统的不断发展，Windows编程也将不断发展。新的 Windows API将不断推出，新的开发工具将不断出现，新的编程技术将不断涌现。

.NET Framework是 Microsoft推出的一个新的开发平台，它提供了一个统一的编程模型，可以用于开发 Windows应用程序、Web应用程序、移动应用程序等。.NET Framework将成为 Windows编程的重要组成部分。

Windows Store应用程序是 Microsoft推出的一种新的应用程序类型，它运行在 Windows 8及以后的版本上，具有良好的用户体验和安全性。Windows Store应用程序将成为 Windows编程的重要发展方向。

八、 结束语

本文介绍了在 Windows下对文件与目录进行编程的基本知识和方法，并给出了一个简单的编程实例。希望读者通过本文的学习，能够掌握 Windows编程的基本技能，并能够开发出自己的 Windows应用程序。

（收稿日期：2008年 5月 15日）



st_gid：文件的用户组的组 ID。

st_size：文件的字节数。

st_atime：文件数据最近一次被访问的时间。

st_mtime：文件数据最近一次被更改的时间。

st_ctime：文件状态最近一次被修改的时间。

opendir：打开一个指定的目录，并返回一个 DIR 类型的指针。DIR 数据类型包含如下两个元素：

d_ino：文件 ID 标识号，如果文件被删除，则 d_ino 为 0。

d_name：此目录下包含的文件名。

readdir：读出指定目录中的一个文件或子目录，并将指针移到下一个文件或子目录。

closedir：关闭相应的目录。

三、 程序说明

该程序是一个类似于 UNIX 命令 ls 的小工具，取文件名为 ls。为了简单起见，此程序只读取了文件的字节数，即 st_size。它可以接受一个参数，指定的目录，也可以不带参数，则默认为当前目录。程序列出指定目录及其子目录中所包含的所有文件的文件名和文件的字节数，并计算出指定目录所占磁盘空间的大小。程序详细说明如下（程序清单见附录）：

第 6 行：包含在程序中要用到的一些头文件。

第 7 行：定义常数 TABSPACES 为 TAB 键所跳过的空格数，用于规整后面显示语句的屏幕格式。

第 8 行：定义一个变量 dir_size 用于存放指定目录所占磁盘空间的大小。

第 9、10 行：判断命令行如果没有带参数，则调用 list 函数，将指定目录置为当前目录，并将返回的目录大小存入变量 dir_size 中。

第 11、12 行：如果命令行带的参数大于一个，则在屏幕上显示出此命令的简要语法并退出程序。

第 13、14 行：如果命令行只带一个参数，则将此参数作为指定目录调用 list 函数，同时也返回目录的大小并存入变量 dir_size 中。

第 15 行：打印出指定目录及其子目录所占磁盘空间的大小。

第 23 行：定义 list 函数，该函数采用递归法列出字符串 name 所指定的目录及其子目录中所包含的文件名和文件的大小，并计算 name 所指定的目录所占用的磁盘空间。

第 25、26 行：定义函数中要用的变量并赋初值。

第 27 行：打印当前所操作目录的目录名。

第 28 行：开始一个循环次数为 1 的 do 循环。第 1 次循环找出目录中包含的所有文件，第 2 次循环找出目录中包含的所有子目录。

第 29、30 行：打开指定的目录，如果打开成功则返回指向此目录的指针，否则返回 NULL，程序打印出出错信息并退出。

第 31 行：开始一个 while 循环，每循环一次读出目录中的一个文件或子目录，直到读出目录中所有的文件或子目录，则返回 NULL，表示已读到此目录的尾部。

第 32、33 行：判断读出的文件的 i 节点标识号，如果为 0，则表示该文件已被删除。程序继续执行并读取下一个文件。

第 34、35 行：在读出的文件名前加上绝对或相对路径，以使程序能正确的从文件系统中访问指定的文件。

第 36、37 行：在 UNIX 下对获取指定文件的信息，将获取的信息存储到结构 stat 中。如果出错则退出程序。

第 38 行：判断读出的文件是否是一个目录，这个条件用语句 if(stat.st_mode&S_IFDIR)来实现。其它三个条件用于滤除符号链接、目录本身和上一级目录，以避免出现死循环。

第 39、40 行：如果是第 2 次 do 循环，则递归调用 list 函数，列出相应的子目录，同时累加目录所占磁盘空间的大小。

第 41、42 行：如果是第 1 次 do 循环，则打印出文件名和文件所占用的字节数，同时累加所占磁盘空间的大小。用三条打印语句是因为文件名长度不一致，为了打印出的屏幕美观。

第 43 行：关闭相应的目录。

第 44 行：返回本目录所占用的磁盘空间。

本程序在 UNIX 操作系统 SCO OpenServer 5.0 和 RedHat Linux 6.0 上调试运行通过。

文件与目录进行编程

近来，随着 UNIX 的映射，由此使得文件系统上的一个结构被作为一个整体对待。一个目录可包含文件，也可以包含子目录，这些子目录同样可以包含更多的文件和子目录。

附录：程序 list.c

```
1  #include <sys/types.h>
2  #include <sys/stat.h>
3  #include <sys/dir.h>
4  #include <string.h>
5  #include <stdio.h>
6  #define TABSPACES 8
7  main(argc, argv)
8  int argc;
9  char **argv;
10 {
11   off_t dir_size;
12   if (argc<2)
13     dir_size =list(".");
14   else if (argc> 2)
15   {
16     printf("Usage: lx [path]\n");
17     exit(1);
18   }
19   else
20     dir_size =list(argv[1]);
21   printf(" This directory size is %i bytes. \n", dir_size);
22 }
23 off_t list(name)
24 char *name;
25 {
26   char pn[255];
27   DIR *dp;
```

在 VisualBasic 环境下编程中提高程序性能的技巧

杜津玲

摘 要 在设计和运行应用程序时，一般都希望应用程序能够发挥最佳性能。本文介绍了在 VisualBasic 环境下编程中提高程序性能的一些技巧。

关键词 VisualBasic 程序，性能，提高

由于 VisualBasic 语言具有强大的性能、丰富而完整的工具、极其友好的图形用户界面、良好的兼容性、独一无二的跨平台特性、无以伦比的速度等特性，所以得到较为广泛的应用。在用 VisualBasic 语言设计和运行应用程序时，一般都希望 VisualBasic 语言及应用程序能够发挥最佳性能。如何才能提高程序性能呢？解决该问题的方法就是对程序进行各种优化。在用 VisualBasic 进行程序设计时，正确、合理地选择使用数据、函数和命令可以达到提高程序性能的目的，现介绍如下：

1.在程序中合理使用 DoEvents 命令。某些关键的 DoEvents 命令对 VisualBasic 的性能影响很大，合理的使用、设置 DoEvents 命令，可以优化 VisualBasic 系统。

DoEvents 命令决定 VisualBasic 是否显示命令结果。在 VisualBasic 中，对话结果可以直接传送到主窗口、图形状态栏或用户自定义窗口中，对话也可以关掉。当 DoEvents 设置为 True 时，VisualBasic 主窗口或用户自定义窗口必须经常更新，因此程序的运行速度会降低。当 DoEvents 设置为 False 时，阻止对话结果传送到主窗口、系统信息窗口、图形状态栏或用户自定义窗口，可以节省更新时间。

DoEvents 命令决定是否在窗口显示执行程序中的命令，当 DoEvents 设置为 True 时，则在窗口显示执行程序中的命令，这就大大降低了速度，将 DoEvents 设置为 False，则当程序运行时不在窗口显示所执行的命令，这就提高了程序的执行速度，同时可避免当程序运行时每次都更新命令窗口。

2.正确合理地选择数据类型。在 VisualBasic 中数据是必不可少的，数据类型的选择对程序执行的速度有一定的影响。对于数值信息，可以选择数值型、浮点型、双精度型和整

```
28  off_t f_size, d_size;
29  int i;
30  struct stat sbuf;
31  struct direct * dir;
32  f_size = 0;
33  printf("Current directory is %s\n", name);
34  for (i = 0; i < = 1; i + + )
35  {
36    if ((dp = opendir(name)) = = NULL)
37    {
38      perror(name);
39      exit(1);
40    }
41    while ((dir = readdir(dp))! = NULL)
42    {
43      if(dir -> d_ino = = 0)
44        continue;
45      strcpy(pn, name);
46      strcat(pn, "/");
47      strcat(pn, dir -> d_name);
48      if (lstat(pn, & sbuf) < 0)
49      {
50        perror(pn);
51        exit(1);
52      }
53       if   (((sbuf. st_mode  &  S_IFMT)! = S_IFLNK)  & &
((sbuf. st_mode& S_IFMT) = = S_IFDIR) & & (strcmp(dir -> d_name, ".")! = 0)& & (strcmp(dir -> d_name, "..")! = 0))
54      {
55        if (i = = 1)
56        {
57          d_size = list(pn);
58          f_size = f_size + d_size;
59        }
60      }
61      else
62      {
63        if (i = = 0)
64        {
65          f_size = f_size + sbuf. st_size;
66          if (strlen(dir -> d_name) > = 2 * TABSPACES)
67            printf("%s\t%i\n", dir -> d_name, sbuf. st_size);
68          else if (strlen(dir -> d_name) > = TABSPACES)
69          printf("%s\t\t%i\n", dir -> d_name, sbuf. st_size);
70          else
71        printf("%s\t\t\t%i\n", dir -> d_name, sbuf. st_size);
72        }
73      }
74    }
75    closedir(dp);
76  }
77  return f_size;
78 }
```

收稿日期：2000 年 3 月 30 日

型，一般情况下尽可能选用整型数据类型，因为 [illegible] 可以最有效地处理整型数据。在建立索引文件时，如果有可能的话，尽量使用整型数据类型字段作为主关键字和外部关键字，这样会得到较小的数据文件，更小的索引，以及更快的连接。

[illegible]避免重复打开文件。重复打开文件会降低性能，较好的作法是将要用到的文件在不同的工作区打开，在需要时使用 [illegible] 命令选择特定的工作区。

[illegible]尽量使用过程文件。一个较复杂的应用系统，通常都有许多的子程序，由于 [illegible] 中程序是以文件形式存储在磁盘上的，因此，每当调用一个外部程序，都要去访问磁盘，而频繁的磁盘存取正是影响程序执行速度的主要因素。要提高程序的执行速度，就应减少磁盘存取次数。[illegible] 提供的过程文件在程序运行期间驻留内存，调用过程文件中的过程时只需发出一个 [illegible] 命令，不需要重新打开这个文件，从而减少了磁盘访问次数，同时也可以减少工作目录中需要的文件数，这样所有文件的操作（如打开、重命名或删除文件）速度均会得到提高。

[illegible]当某一函数或运算需要重复多次时，把这个函数和运算结果保存在一个内存变量中，以后就可以用这个内存变量来代替这个函数或运算，这样可以节省一些时间。

[illegible]合理使用循环语句。在 [illegible] 中有三种循环语句：

循环：[illegible]

自动计数循环：[illegible]

扫描数据库循环：[illegible]

一般情况下，当循环次数已知时，循环既可以使用 [illegible]，也可以使用 [illegible]，应尽量使用 [illegible] 循环。例如：从 [illegible]

用 [illegible] 循环程序如下：

```
S=0
FOR N=1 TO 100
S=S+N
ENDFOR
```

用 [illegible] 循环程序如下：

```
S=0
N=1
DO WHILE N<=100
S=S+N
N=N+1
ENDDO
```

从上例可以看出，用 [illegible] 循环，循环变量的初值可不用先设定，累计也不需另设 [illegible]，因此，在计数循环时 [illegible] 比 [illegible] 更快，且程序更简洁。

若对数据表中全部记录或符合条件的记录执行某一操作时，使用 [illegible] 循环较好。例如：逐条显示数据表中的记录。

用 [illegible] 循环：

```
USE <数据表名>
SCAN
DISPLAY
ENDSCAN
```

用 [illegible] 循环：

```
USE <数据表名>
DO WHILE NOT EOF()
DISPLAY
SKIP
ENDDO
```

可见 [illegible] 不需要 [illegible] 命令逐一控制记录指针，也不需用 [illegible] 判断文件是否终结。因此，在数据表循环读取时，[illegible] 比 [illegible] 速度快。

[illegible]要注意循环内各种操作的效率问题。如：循环内不变的运算尽量提到循环外，削弱循环内运算强度等，可提高运行效率，特别是循环次数很大时。

[illegible]用名称表达式代替宏替换。[illegible] 中的宏替换函数用途很广，灵活、巧妙地使用能增加程序的灵活性、通用性、减少程序量。在有些情况下，用名称表达式来取代宏替换，程序性能将有很大改进。

在程序设计中，考虑到程序的通用性，一般习惯用下列语句打开一个数据表

```
ACCEPT "输入数据表名称" TO MC
USE &MC
```

若将 [illegible] 改为 [illegible]，即宏替换改为名称表达式，则更快。

[illegible]尽量将输出发送到顶层窗口，将输出发送到任何非顶层窗口都比输出到顶层窗口慢。使输出在窗口后面滚动是最糟糕的情况，应尽量避免。

[illegible]使用数组，可以使数据的处理及其管理变得简便有效。引用数组，只要改变其下标变量即可，从而易于用循环结构进行处理，大大减少程序量，使程序性能得以改善。从多个字段复制数据时，[illegible] 命令要比 [illegible] 命令快。

[illegible]高效地引用多个属性。如果更新对象的多个属性，[illegible] 必须多次寻找这个对象，这会影响程序的性能。为了避免这种重复查找，可使用 [illegible] 命令。这种方法使 [illegible] 只查找一次对象。

[illegible]采用动态内存管理，在运行时能动态地调整内存分配，使内存能得到有效地利用。当 [illegible] 运行在内存较大的环境下时，应用程序将能分配到较多的空间，可提高程序的运行速度。若要最有效地使用内存，尽量在需要对象时创建，当不再使用对象时要及时清除，以便释放内存。若要检查每个对象占用了多少内存，可调用 [illegible] 函数。

收稿日期：[illegible]年[illegible]月[illegible]日）

网页中多层效果的灵活使用

李 莹

动态的网页技术 [illegible] 已经非常普及，本文强力向大家推荐一段完整并且是经典的网页多层代码，适用于 [illegible] 及 [illegible] 游览器，可实现完善的二层网页功能，具体代码请见文后程序。

该段代码功能：打开该网页文件之后，将在正常的网页之上浮动另外一层网页内容，可用鼠标随意移动第二层网页在第一层上的位置，并且当屏幕滚动时，第二图层可始终保持同第一图层的相对位置，效果非常好。

适用场合：有了这段代码，你可以对很多原有的网页进行改造，使游览者更为方便，比如可用于留言簿，第一层放置留言内容，浮动的第二层放置留言表单，这样游览留言操作和观察留言内容可在同一页面完成，并且无论屏幕滚动到什么位置，留言表单始终出现在屏幕之上，这无疑是最方便的留言系统工作方式。再比如可做为一般网页的导航条，有很多人为了编制美观的页面而放弃使用分栏导航页面方式，而使用整个页面，这样的页面每进入到一个分栏目，都要重新调入整个页面。有了这项技术，可以在原完整的页面之上，浮动一层导航菜单，既方便了导航又保持了页面的完整。另外也可以在某些页面之上浮动一个特定的图像或其它页面内容，等等，总之，这种技术应用非常广泛。

代码说明：代码段一可放置在网页的文件头处，即 [illegible] 段。代码段二为正常的网页内容，即第一层的代码内容。第三段为第二层网页的代码段，即浮动的网页内容。代码段四为必要的控制代码，可放置在网页的最后处。

灵活使用：下面的代码是完整的网页文件代码，把代码段二和代码段三换成你自己所需要的内容就可以了，但由于本网页文件涉及两个网页的制作过程，可能制作起来相当费力，并且很多网页制作工具不支持这一功能，用低级的网页工具存盘后将失去浮动效果，所以对于这样的网页最好的维护方法是分页维护，即分别编制两个完整的网页内容，之后按要求合并在一个网页文件内即可。这样向你介绍一种最可取的方法，众所周知，现在的网页技术可以实现在一个网页内调用其它网页的功能，即所谓的画中画功能，所以完全可以把代码二和代码三换成固定的调用其它网页的固定代码，之后只需要对这两个单独的页维护就可以了。具体可用下面的代码替换代码二和三：

```
ILAYER id = ″ad1″ visibility = ″hidden″ height = ″60″>
 </ILAYER>
 <NOLAYER>
 <IFRAME SRC = ″ h1. htm″ width = ″ 100%″ height = ″ 60″
marginwidth = ″0″ marginheight = ″0″ hspace = ″0″ vspace = ″
0″ frameborder = ″0″ scrolling = ″no″>
```

[illegible] 其中 [illegible] 用于指定打开的网页地址

[illegible] 表格占用的宽度

[illegible] 表格点用的高度

[illegible] 网页内容在表格右侧的预留宽度

[illegible] 网页内容在表格顶部预留的高度

[illegible] 网页右上角的横坐标

[illegible] 网页右上角的纵坐标

[illegible] 是否显示边缘

[illegible] 是否出现滚动条

具体使用时把 [illegible] 分别换成不同的网页文件名。

完成的网页文件代码如下：

```
 <html>
 <head>
 <meta http – equiv = ″ Content – Type″ content = ″ text/html;
charset = gb2312″>
//代码段一
 <style type = ″text/css″> <! – –
#floater {position: absolute; left: 500; top: 146; width: 125;
visibility: visible; z – index: 10; }
      – – >
 </style>
//代码段一结束
 <title> 浮动层 </title>
 </head>
//代码段二(第一层内容,目前为空)>>>>
 <body bgcolor = ″#FFFFFF″>
 <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR
 <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR
 <BR> <BR> <BR> <BR> <BR> <BR> <BR>
 <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR
 <BR> <BR> <BR> <BR> <BR> <BR> <BR> <BR
 <BR>
//代码段二结束
//代码段三(浮动层内容,目前为一简单表格)/
 <div ID = ″floater″ style = ″left: 0px; top: 1px″>
 <table border = ″1″ cellpadding = ″0″ cellspacing = ″0″
bgcolor = ″#25A78D″ bordercolordark = ″#00FFFF″
bordercolorlight = ″#000000″>
    <tr>
       <td> 111 <a       href = ″       file: ///G: /C/NEW
dhtml2. htm″> 1111 </a> </td>
       <td> 2222222 </td>
    </tr>
 </table>
 </div>
//代码段三结束
//代码段四
 <script LANGUAGE = ″JavaScript″>
      self. onError = null;
```

```
currentX = currentY = 0;
whichIt = null;
lastScrollX = 0; lastScrollY = 0;
NS = (document.layers) ? 1 : 0;
IE = (document.all) ? 1: 0;
<!-- STALKER CODE -->
function heartBeat() {
if(IE) { diffY = document.body.scrollTop; diffX = document.body.scrollLeft; }
if(NS) {diffY =self.pageYOffset; diffX = self.pageXOffset; }
if(diffY != lastScrollY) {
percent = .1 * (diffY - lastScrollY);
if(percent > 0) percent = Math.ceil(percent);
else percent = Math.floor(percent);
if(IE) document.all.floater.style.pixelTop += percent;
if(NS) document.floater.top += percent;
lastScrollY = lastScrollY + percent; }
if(diffX != lastScrollX) {
percent = .1 * (diffX - lastScrollX);
if(percent > 0) percent = Math.ceil(percent);
else percent = Math.floor(percent);;;; {
if(IE) document.all.floater.style.pixelLeft += percent;
if(NS) document.floater.left += percent;
lastScrollX = lastScrollX + percent; }}
<!-- /STALKER CODE -->
<!-- DRAG DROP CODE -->
function checkFocus(x,y) {
stalkerx = document.floater.pageX
stalkery = document.floater.pageY
stalkerwidth = document.floater.clip.width;
stalkerheight = document.floater.clip.height;
if( (x > stalkerx && x < (stalkerx+stalkerwidth)) && (y > stalkery && y < (stalkery+stalkerheight))) return true;
else return false; }
function grabIt(e) {
if(IE) {whichIt = event.srcElement
while (whichIt.id.indexOf("floater") == -1) {
whichIt = whichIt.parentElement;
if (whichIt == null) { return true; } }
whichIt.style.pixelLeft = whichIt.offsetLeft;
whichIt.style.pixelTop = whichIt.offsetTop;
currentX = (event.clientX + document.body.scrollLeft);
currentY = (event.clientY + document.body.scrollTop);
} else {
window.captureEvents(Event.MOUSEMOVE);
if(checkFocus (e.pageX,e.pageY))
whichIt = document.floater;
StalkerTouchedX = e.pageX - document.floater.pageX;
StalkerTouchedY = e.pageY - document.floater.pageY;
} }
return true;}
function moveIt(e) {
if (whichIt == null) { return false; }
if(IE) {
newX = (event.clientX + document.body.scrollLeft);
newY = (event.clientY + document.body.scrollTop);
distanceX = (newX - currentX);
distanceY = (newY - currentY);
currentX = newX; currentY = newY;
whichIt.style.pixelLeft += distanceX;
whichIt.style.pixelTop += distanceY;
if(whichIt.style.pixelTop < document.body.scrollTop) whichIt.style.pixelTop = document.body.scrollTop;
if(whichIt.style.pixelLeft < document.body.scrollLeft) whichIt.style.pixelLeft = document.body.scrollLeft;
if(whichIt.style.pixelLeft > document.body.offsetWidth - document.body.scrollLeft - whichIt.style.pixelWidth - 20) whichIt.style.pixelLeft = document.body.offsetWidth - whichIt.style.pixelWidth - 20;
if(whichIt.style.pixelTop > document.body.offsetHeight + document.body.scrollTop - whichIt.style.pixelHeight - 5) whichIt.style.pixelTop = document.body.offsetHeight + document.body.scrollTop - whichIt.style.pixelHeight - 5;
event.returnValue = false;
} else {
whichIt.moveTo(e.pageX - StalkerTouchedX, e.pageY - StalkerTouchedY);
if(whichIt.left < 0 +self.pageXOffset) whichIt.left = 0 +self.pageXOffset;
if(whichIt.top < 0 +self.pageYOffset) whichIt.top = 0 +self.pageYOffset;
if((whichIt.left +whichIt.clip.width)> =(window.innerWidth +self.pageXOffset -17))whichIt.left =((window.innerWidth +self.pageXOffset) -whichIt.clip.width) -17;
if((whichIt.top +whichIt.clip.height)> =(window.innerHeight +self.pageYOffset -17))whichIt.top =((window.innerHeight +self.pageYOffset) -whichIt.clip.height) -17;
return false; } return false; }
function dropIt() {
whichIt = null;
if(NS) window.releaseEvents (Event.MOUSEMOVE);
return true; }
<!-- DRAG DROP CODE -->
if(NS) {
window.captureEvents(Event.MOUSEUP|Event.MOUSEDOWN);
window.onmousedown = grabIt;
window.onmousemove = moveIt;
window.onmouseup = dropIt; }
if(IE) {
document.onmousedown = grabIt;
document.onmousemove = moveIt;
document.onmouseup = dropIt; }
if(NS || IE) action = window.setInterval("heartBeat()", 1);
</script>
//代码段四结束
</body>
</html>
```

穴收稿日期 年 月 日

Windows 98下的多显示器编程技术

王 筱

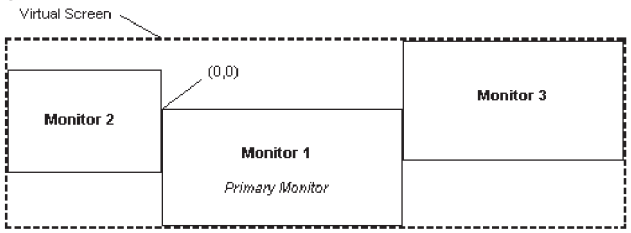
摘 要 文介绍在 宰蜘蛛精释怨愿操作系统下对多个显示器编程的方法,并提供了一个枚举每一个显示器和显示设备信息的程序实例。

关键词 宰蜘蛛精释怨愿虚拟屏幕,动态链接库 闭蛋蓝

一、关于多显示技术的概述

微软公司在其推出的操作系统 宰蜘蛛精释怨愿中提供了对多个显示器的支持，根据微软的文档 宰蜘蛛精释怨愿支持最多达 怨个的显示器。这就我们计算机用户带来了很大的方便，特别是对程序员来说 宰蜘蛛精释怨愿的这个特性可以使我们调试程序变得更加方便。但是目前的许多应用程序都不能很好的支持多个显示器，如何使你的程序能支持 宰蜘蛛精释怨愿下的多个显示器呢？这就是本文将要向大家介绍的。

首先，介绍一下在 宰蜘蛛精释怨愿下的多显示器是如何工作的。为了支持多个显示器微软提出了虚拟屏幕的概念，即整个桌面可以看成是一个虚拟屏幕，一个虚拟屏幕由多个监视器组成。下图是由三个监视器组成的一个虚拟屏幕的一种可能的布局：



Monitor Topology

其中虚拟屏幕的原点为第一显示器 缸显示器）的左上角，其它显示器的坐标都是相对于第一显示器的。关于如何安装和设置多个显示器已有多篇文章介绍，在这里我们就不多说了。

二、多显示器编程接口

接下来我们介绍一下多个显示器编程的软件接口。微软为了支持多个显示器的编程微软在 宰蜘蛛精释怨愿中增加入了以下接口函数，这些函数的实现在 哉猜祖蜘蛛精蓝中：

EnumDisplayDevices( )
EnumDisplayMonitors( )
GetMonitorInfo( )
MonitorFromPoint( )
MonitorFromRect( )
MonitorFromWindow( )

并新增了一个新的句柄：匀酝诸猿酿鲫狈

我们先来认识一下 匀酝诸猿酿鲫狈 句柄 匀酝诸猿酿鲫狈是一个猿圆位的整数用来表示一个物理上的显示设备。为了得到或设置一个显示器，你必须首先拥有个的 匀猿诸猿酿鲫狈句柄，并使这个句柄总是指向同一个显示器。

以下是各个 粤孕滥函数的简单说明：

要对有多显示器的计算机进行编程首先要检测这台计算机上安装了几台显示器，函数 耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽 和 耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽耘曾 就是用来枚举每台显示器的信息的。

```
BOOL EnumDisplayMonitors (
HDC hdc,                    // handle to a display device context
LPCRECT lprcClip,           // specifies a clipping rectangle
MONITORENUMPROC lpfnEnum, //pointer to a callback function
LPARAM dwData       // data passed to the callback function
);
```

调用这个函数后系统将自动调用 遭藻泽藻责造葬赠酝燥灶蚤贼燥则泽 指向的回调函数

```
BOOL CALLBACK MonitorEnumProc(
HMONITOR hMonitor,   // handle to display monitor
HDC hdcMonitor, //handle to monitor –appropriate device context
LPRECT lprcMonitor, //pointer to monitor intersection rectangle
LPARAM dwData  // data passed from EnumDisplayMonitors
);
```

耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽 中的 澡酝燥灶蚤贼燥则 遭藻泽藻责造葬赠酝燥灶蚤贼燥则泽 将传递给 配燥造造遭葬糟噪 函数 耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽 中的 澡酝燥灶蚤贼燥则 遭藻泽藻责造葬赠酝燥灶蚤贼燥则泽 其中 澡酝燥灶蚤贼燥则 是你所感兴趣的显示设备句柄，遭藻泽藻责造葬赠 是你所感兴趣的区域的指针，凿葬贼葬 是附加数据。如果想要枚举所有的显示器可以用如下代码完成：耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽（晕哉蚤灶，晕哉蚤灶，配燥造造遭葬糟噪，耘灶哉蚤泽藻酝燥灶蚤贼燥则泽责则燥糟，凿葬贼葬）；

系统中安装了 灶个显示器回调函数就会被调用 灶次，依次在 澡酝燥灶蚤贼燥则 参数中给出每个显示器的 匀酝诸猿酿鲫狈 这样在回调函数中就可以得到每个监视器的信息。

耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽 函数能得到每个监视器的句柄，这样我们就可以使用 郧藻贼酝燥灶蚤贼燥则陨灶枣燥 函数来取得各显示器的具体信息。

```
BOOL GetMonitorInfo(
HMONITOR hMonitor,   // handle to display monitor
LPMONITORINFO lpmi//pointer to display monitor information
);
```

郧藻贼酝燥灶蚤贼燥则陨灶枣燥 函数将返回句柄为 澡酝燥灶蚤贼燥则 的监视器的信息，这些信息返回在 酝燥灶蚤贼燥则陨灶枣燥 变量的 或 酝燥灶蚤贼燥则陨灶枣燥耘曾 结构中。显示器信息包括显示器范围，显示器工作范围，第一显示器 缸显示器）标志及监视器名称。

耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽 只能枚举监视器，而 耘灶哉蚤泽藻糟则陨藻责造葬赠酝燥灶蚤贼燥则泽耘曾 能枚举显示设备，最典型的显示设备就是显示适配器。

需要注册请访问 www.ertongbook.com