神州易刻二次开发资料(创造附加业务价值)

前言：(因为我个人很忙，恕不提供任何咨询与协助服务，请知悉！)

神州易刻 2013.02 版本已经 11 岁半了！这么多年我们一直没空升级它，它非但没有被超越，反而奇迹般地创造了惊人的用户黏性。我们 2024 年 1 月发布神州易刻 2024.01 版本后，一些老用户甚至被新版本神州易刻的巨大变化吓了一跳(突然觉着空荡荡的)！但事实上，新版本神州易刻相比老版本神州易刻，不仅仅是增强增加了很多功能，而且其易用性更是大幅度提升。只是我们一直没有时间写个基础教程展示其新特性，但可以肯定的是：只要用户基本习惯了新版本神州易刻，你一定不愿意再回去用古老的神州易刻 2013.02！

神州易刻 2024.01 版本发布前跟一个设备制造厂家的对话。一个 11 年都不曾升级一次的软件竟然获得了 95%的用户支持，且不管 95%这个数据是否真实，但至少从侧面说明了神州易刻具有极强的用户黏性

可能因为神州易刻有极其卓越的用户黏性，于是就出现了一些矛盾：比如某公司承接了一个项目，但该公司自行编写的软件，无法为用户所接受，寻求我们帮助而我们又没有时间去解决。再比如某些情况下，需要一些特别的功能(比如脱机作业)，而我们的控制主板暂时未提供该功能，而使用其他主板制造设备，因为无法使用神州易刻直接输出，用户又不愿意接受。此外，因为各地备案系统不同，神州易刻如何接口不同的备案系统？还有一些备案系统是把排版出来的图样交付给刻制公司刻制，而备案系统排版的图样，因为其排版设计者可能并不了解激光刻制的技术内幕，致使其提供的图样刻制效果难以达标(比如笔画过细、图样 dpi 不匹配设备 dpi)……

如果神州易刻不提供二次开发功能，那么要解决这些问题，都只能找我们编写相应的功能。但是，我大多数时候是没有时间也没有兴趣去处理这样一个个的零碎需求。另一方面，因为我们所提供的功能都是针对所有厂家和用户的，所以不利于第三方创造附加的业务价值。

为了解决你的业务需求需要我协助，而我没有时间响应你的需求的矛盾，新版本神州易刻提供了非常丰富的二次开发功能，任何第三方都可以利用神州易刻的二次开发功能去创造附加的业务价值，但是你必须自主完成二次开发，我们没时间协助。下面我们就看看安装包仅 2.35M 的新版本神州易刻提供了多少二次开发功能吧！古董版本的神州易刻 2013.02 的安装包是 3.01M。

2024.06.04
李辉宇
杭州宇骐科技有限公司

ini 文件脚本控制

神州易刻有两个 ini 文件： RunOptions.ini 和 EngraverSetting.ini，可以在这两个文件里添加一些预定义的控制字对神州易刻的运行和激光雕刻机进行一些特殊的控制或配置。这两个 ini 文件的定位位置在{CSIDL_COMMON_APPDATA}\Lihuiyu Studio Labs\LaserDRW\，这个位置应该使用 Windows API SHGetFolderPath 先定位 CSIDL_COMMON_APPDATA 路径，不能简单地在你的计算机上复制你的计算机上的字符串路径，因为在不同的 Windows 操作系统里，CSIDL_COMMON_APPDATA 路径字符串并不一定是相同的，而且用户是可以自己更改其位置的，所以要兼容各种情况，应使用 API SHGetFolderPath 获取。

约定：以下类似 xxxx=[0, 1]，表示某个控制字可取值 0 或 1，蓝色 1 则表示默认值为 1。默认值即 ini 文件里没有写入这个控制字，神州易刻就自动设置为默认值。

RunOptions.ini

[Dll Params]

BinColorImg=[0，1]　　范例：BinColorImg=0

{BinColorImg：控制人工导出 Windows Bitmap 格式图象时，是真彩色还是二值彩色。二值彩色占用的存储空间约为真彩色的 1/24(跟黑白位图完全一样)，但它不只是黑白，还可以是红白、蓝白、黄白、(任意色)白。考虑到印模实质只有两种颜色，二值彩色图足够使用，故提供了二值彩色位图导出功能。图象识别技术大多数时候都要先转换为二值图，也就是说使用二值图作为印模的标准格式，会更适合开发或接口印模的自动识别技术

}

[Scripts]　　//第一组：控制雕刻时是否自动导出图像

SaveEgvDir=[string]　　范例：SaveEgvDir=C:\我的工作文件夹\

{SaveEgvDir：设置启动雕刻后，是否自动导出雕刻的图象。该字段设置雕刻时自动导出的图象存储的根目录，根目录下会依据日期自动创建子目录，比如{SaveEgvDir}\ 2023-12-29\，日期子目录下存储当天所导出的(也即所雕刻的)所有图象。而图象文件名则以导出的时间为名，文件名前缀有两种：A.和 N.(因为可设置同时导出两张图象)，比如文件名 A.12-32-16.png 是包含透明度 alpha 通道的 png 图象，而 N.12-32-16.xxx 为指定格式图象(xxx 为 bmp png jpg gif emf tiff pcx 之一)。SaveEgvDir 是总开关，以下控制字依赖这个总开关是否有效

}

Resolution=[0 - 4000]　　范例：Resolution=1000

{Resolution：设置启动雕刻时自动导出图象的分辨率(dpi)，Resolution 应为整数，默认值是 600

}

FormatIndex=[0 - 6]　　范例：FormatIndex=2

{FormatIndex：设置启动雕刻时自动导出图象的格式，0 至 6 依次对应的图象格式：bmp、png、jpg、gif、emf、tiff、pcx。比如 FormatIndex=1，表示导出 png 图象

}

Monochrome=[0, 1]　　范例：Monochrome=0

{Monochrome：设置是否导出单色图象。若 FormatIndex=0 (即 bmp)，也可导出二值彩色位图，二值彩色

位图的颜色由下面的 Rendering 控制字指定的颜色决定
}

Rendering=[0xBBGGRR] 范例：Rendering=255 //即红色
{Rendering：控制导出的图象渲染上什么颜色。应为十进制整数，不可用 0xBBGGRR 样式，此样式仅为了方便理解 RGB 颜色格式。默认颜色是 0x002828E6(朱砂印泥色)
}

PngAlpha =[0 - 255] 范例：PngAlpha=128
{PngAlpha：如果 FormatIndex=1(即 png)时，该控制字控制导出的 png 图象的透明度。应注意的是：为方便理解，该值与理论值是相反的，Alpha 通道理论上 0 表示全透明，而我们设置为 255 表示全透明，也就是说：PngAlpha 实际等于理论的 255- PngAlpha，这样做的目的是为了符合理解习惯：透明值越大越透明。PngAlpha 默认值是 96
}

ClipImage=[0, 1] 范例：ClipImage=0
{ClipImage：控制导出图象时，是否修剪掉四边多余的空白。ClipImage=1 表示修剪掉空白
}

OutAlphaPng=[0, 1] 范例：OutAlphaPng=1
{OutAlphaPng：控制是否同时导出一张可设置透明度的 alpha png 图象(可用于电子印章)，比如我们设置控制字 FormatIndex=0(即 bmp)，还设置了 OutAlphaPng =1，那么导出时会同时导出两张图片：A.xx-xx-xx.png(用于电子印章)和 N.xx-xx-xx.bmp(用于上传到备案系统)
}

SendClipbrd=[0, 1] 范例：SendClipbrd=1
{SendClipbrd ：控制导出图象时是否把图象句柄(HBITMAP)写入到系统剪贴板，否则把导出图象的文件名写入系统剪贴板。该功能主要方便插件或其他中间件处理导出的图象，比如可监视系统剪贴板的变化，从剪贴板读取 HBITMAP 或存储的文件名，经插件或中间件二次处理后上传到备案系统。组版页面因为排版有多个图形，所以导出时是有多个图形，此时系统剪贴板里写入的是多个文件名，文件名以 | 分割。
提示：如果有外部协作软件，神州易刻导出图片时会及时通知外部协作软件，不用监视系统剪贴板。详见后文的说明
}

Comfirm=[string] 范例：Comfirm=已开始雕刻，需要导出正在雕刻的图象吗?
{Comfirm：因为可能并不是每次雕刻都需要导出，设置一个字符串比如雕刻开始了，你想导出正在雕刻的图象吗? ，如果没有设置该控制字，则不会请求用户确认，直接导出
}

提示：以上控制字(Resolution\ FormatIndex\ Monochrome\ Rendering\ PngAlpha\ ClipImage\ OutAlphaPng\ SendClipbrd\ Comfirm)必须在设置有效的 SaveEgvDir 文件夹才有效，也就是说 SaveEgvDir 为总控开关，其他与之相关的控制字，均要受控于 SaveEgvDir 的设置！ 应用：刻制前我们并不能确定刻制出来是否合格，此时我们不能上传到备案系统，确认刻制合格后我们才上传到备案系统，但刻制过程我们可能还要继续编辑与排版。所以设置雕刻时立即导出图象，那么我们等雕刻完成确认合格即可上传导出的图象。

[Scripts] //第二组：仍在该字段下，仅为阅读方便。以下一组控制主要控制外部程序同步协作

StartSyncApp=[string] 范例：StartSyncApp=D:\神州易刻增强\YKSync.exe

{StartSyncApp：外部协作程序的 exe 文件路径(我们建议是 32bit 应用程序)，当该控制字为有效 exe 文件时，神州易刻启动时会同步启动该程序，并与该程序建立联系。所以，你可以编写一个独立的外挂到神州易刻的程序，借助神州易刻干你自己想做的事情，比如上传印模、挂接其他的雕刻机雕刻神州易刻设计的图象，随你自己发挥

}

CloseSyncApp=[0, 1, 2] 范例：CloseSyncApp=2

{CloseSyncApp：该控制字控制神州易刻关闭时，如何处理外部协作程序

CloseSyncApp=0：不做任何处理

CloseSyncApp=1：如果是神州易刻启动时同步启动的，神州易刻关闭时也关闭它

CloseSyncApp=2：不管是谁启动的，神州易刻关闭时强制关闭它

}

SyncAppExitHint=[string] 范例：SyncAppExitHint=同步协作程序异常，要重新启动并联系它?

{SyncAppExitHint：神州易刻尚未关闭，若因为用户误操作或其他原因，导致外部协作程序意外关闭(或其窗口 HWND 被销毁重建)，提示用户如何处理。比如设置 SyncAppExitHint =同步协作程序已关闭，是否需要重新启动并联系它? 若不设置该控制字，则不做任何提示

}

WndMessage=[大于 WM_USER(即 1024)的整数] 范例：WndMessage=1025

{WndMessage：自定义一个 Windows 窗口消息 ID，StartSyncApp 主窗口应能接收与处理该消息。该消息是外部协作程序 StartSyncApp 与神州易刻互动协作的关键。该消息的解析如下：

WParam：神州易刻主窗口的句柄 HWND，StartSyncApp 的消息处理函数应判断消息来源是否是来自神州易刻，防止消息并非神州易刻所发出，造成错误动作，比如：
if Wparam==YKWnd ……

LParam：低 8 位

0：神州易刻发出的联络信号(LParam=0 仅表示请求联络)，此时应保存一份 WParam(神州易刻主窗体的句柄)。比如保存为 YKWnd=WParam，方便后续判断消息来源是否是神州易刻
提示：神州易刻的主窗口的句柄也可能发生变化！神州易刻的主窗口句柄发生变化时，会重新联络 StartSyncApp，此时仍要执行 YKWnd=WParam

1：神州易刻已启动雕刻

2：神州易刻雕刻完成(LParam =0x10002：正常完成，LParam =0x00002：异常结束)

3：神州易刻停止雕刻(即用户主动停止)

4：神州易刻人工导出了图象(即用户自己执行了导出图象的操作)
LParam = 0x20000004：HBITMAP 在剪贴板
LParam = 0x00000004：文件名在剪贴板

5：神州易刻启动雕刻时自动导出了图象
LParam = 0x20000005：HBITMAP 在剪贴板
LParam = 0x00000005：文件名在剪贴板

6：神州易刻监听到监听目录有新文件并创建了新的图形
LParam = 0x10006：新图形在组版页面

LParam = 0x00006：新图形在设计版面

}

备用方案：神州易刻启动时会同步启动 StartSyncApp，并联系它，但可能因为安全机制限制而无法联络(一般都可以联系上)，那么可设置以下两个控制字用于补救式联络

WndClassName=[string] 范例：WndClassName=ClassYKSyncWnd

{WndClassName：StartSyncApp 主窗口的类名

}

WndCaption=[string] 范例：WndCaption=YKSync

{WndCaption：StartSyncApp 主窗口的 Caption，一般建议不要使用完整的 Caption，只要取其中部分关键字即可，避免版本更新或其他原因导致 Caption 全名变化而不兼容，比如 StartSyncApp 程序的主窗口 Caption 是 YKSync Ver10，可设置 WndCaption =YKSync

}

[Scripts] //第三组：用于排版自动化，其他软件把文件存放到监听目录，神州易刻自动排版

ListenDir=[string] 范例：ListenDir=C:\我的工作\易刻监听\

{ListenDir：指定神州易刻监听的文件夹路径。会同时监听该文件夹下的所有子文件夹，方便中间件设计者归类组织自己的数据。当监听目录(或其子目录)下有符合要求的文件(bmp png jpg jpeg gif emf wmf tiff pcx yzi)到达监听文件夹(或其子文件夹)里，将在设计版面或组版版面自动创建一个或多个图形。bmp png jpg jpeg gif emf wmf tiff pcx 文件名格式化说明：

文件名格式化规则：#n40.0%50.4%L20.6%Tabcd.bmp

#n 或&n：#表示在设计版面创建图形，& 表示在组版版面创建图形，n=1 - 7，对应扫描区为矩形、椭圆、菱形、下三角形、上三角形、左三角形、右三角形

40.0%指定版面宽度是 40.0mm(不限定 3 位，可以是任意位，比如 240.3%)，高度会自动依据图象文件的宽高比自动设置

50.4%L20.6%T，如果是在组版页面创建图形，那么创建的图形左上位置定位在 Left=50.4mm，Top=20.6mm。如果是在设计版面创建，则不用该项格式化

提示：yzi 的文件名无需格式化，它不是图象格式的文件

}

ListenCreateTo=[0, 1] 范例：ListenCreateTo=1

{ListenCreateTo：如果文件名未被格式化，或不符合格式化规则，那么该控制字指定监听到没有格式化的文件时，默认创建图形到设计版面还是组版版面

ListenCreateTo =0：在设计版面创建

ListenCreateTo =1：在组版版面创建

提示：图象文件名未被格式化时，神州易刻会弹出版面设置窗口，由人工设置版面

}

AutoSaveDir[string] 范例：AutoSaveDir=C:\我的工作\

{AutoSaveDir：自动排版创建的文件，可以指定排版后自动保存到什么文件夹里

}

/**/： bmp png jpg jpeg gif emf wmf tiff pcx yzi：除支持的图象文件外，还有 yzi 这个扩展名的文件，该

文件套用指定的模板创建图形，并用指定的文字直接替换掉模版内部的文字。yzi 文件也就是设置如何自动套用指定的模板

::yzi 文件的格式说明(开始)

[Setting] //注意现在是讲述 yzi 文件格式，实际它也是标准 ini 文件，只是扩展名为.yzi

CreateToPage = [0，1] 范例：CreateToPage=1

{CreateToPage：指定套用模板时，图形创建到设计版面还是组版版面

CreateToPage =0：创建到设计版面

CreateToPage =1：创建到组版版面

}

ClearPage =[0，1] 范例：ClearPage=1

{ClearPage：指定是否清空设计版面(或组版版面)

ClearPage =0：版面保持原样

ClearPage =1：清空当前版面

}

SaveAs=[string] 范例：SaveAs=宇骐科技 40 公章

{SaveAs：套用模板的图形创建后，设计版面(或组版版面)以什么文件名保存版面，如果文件名为空，则由用户手动保存。文件名可以省略扩展名，神州易刻会自动添加，比如 SaveAs = 我的\abc，如果设置了 AutoSaveDir ，那么文件实际保存为{AutoSaveDir}\我的\abc. lyz(或 pyz)

}

[New design0] //图形 0

TempIName=[string] 范例：TempIName=c:\我的模板\名章 20.tyz

{TempIName：指定套用的模板文件名。比如 TempIName =c:\我的模板\公章 40mm.tyz

}

神州易刻激光雕刻测试样品=宇骐科技激光雕刻排版测试软件

测试用模板=合同专用章

0123456789210=0779876543210

说明：假定 TempIName 指定的模板里有神州易刻激光雕刻测试样品、测试用模板、0123456789210 三个文字元素，简单照上面的 XXXX=YYYYY 规则，就会直接替换文字，但不会影响原有的模板文件

Left =[xx.n] 范例：Left=23.5

Top=[yy.m] 范例：Top=16.9

{Left、Top：如果图形是创建到组版版面，那么创建的图形左上位置在版面上的坐标(单位 mm)：

Left = xx.n

Top=yy.m

}

[New design1] //图形 1

……

[New designN] //图形 N

{ New design1……New designN：如果是创建到组版版面，那么可依据 New design0 的一样的规则，在设计版面以套用模板的方式创建多个图形

这个功能有什么作用? 因为备案系统可以不需要排版功能，只需要发送一份符合要求的 yzi 文件到神州易刻的监听目录即可。神州易刻排版出来的会比备案系统所排版的更适合激光刻制或打印机打印！所以我们建议把排版工作交给神州易刻来完成可能会更好一些！备案系统一般是交付(bmp 或 jpg 或 png)位图图片的方式交付给刻制者，但是交付的图片并不是适配了激光雕刻机或打印机的 dpi，所以在激光刻制或打印机打印时需要进行缩放以匹配设备(激光雕刻机或打印机)的 dpi，这会导致效果变差，因为非矢量的位图缩放是会丢失精度的。但神州易刻是完全匹配设备精度的输出，具有最完美的效果!

}

::yzi 文件的格式说明(结束)

[Scripts] //第四组：用于控制使用体验与方便性：继续说明 RunOptions.ini

AutoClipOut=[0, 1] 范例：AutoClipOut=0

{AutoClipOut：组版雕刻多个图形时，并不一定排满版面，AutoClipOut 控制是否裁切掉未排版的右下空白区域(但左上空白区域不裁切，因为左上作为精确基准点，避免裁切后多次取整运算，导致切割位置精准度降低)。裁切后直观性下降，若用户方向判别能力比较差，也可以设置 AutoClipOut =0，那么组版雕刻时是原样映射整个页面(比如用户是排版在 A4 纸上，那么进入雕刻管理器时，预览图也是 A4 纸的图样)，直观性比较好

}

AutoCheckTextLang=[0, 1] 范例：AutoCheckTextLang=0

{AutoCheckTextLang：文字输入时，是否自动检测输入文字的语种，这样，除直排的蒙文外，大多数语言文字都可以当作中英文一样排版，因为神州易刻会自动检测语种，依据语种选择合适的排版方式。比如在神州易刻 2013.02 版本中，排版维文要使用……维文或其他连体文字进行排版，而新版本中，维文也可以象中英文一样输入排版

}

AltOpenMenu=[0, 1] 范例：AltOpenMenu=0

{AltOpenMenu：按键盘上的 Alt 键，是否展开主菜单

AltOpenMenu =0：Alt 键与主菜单无关

AltOpenMenu =1：按 Alt 键展开主菜单

}

EdgeEvents=[0, 1] 范例：EdgeEvents=0

{EdgeEvents：边沿事件使能：双击左边沿，打开/关闭标尺；双击右边沿，打开/关闭属性面板。

EdgeEvents =0：屏蔽边沿事件

EdgeEvents =1：使能边沿事件

}

szTemplThumb=[0 - 6] 范例：szTemplThumb=4

{szTemplThumb： 模板文件里缩略图的重取样倍数。重取样倍数越高，模板缩略图越清晰，但文件存储尺寸越大。该设置主要考虑兼容高分辨率屏幕

}

CustomCursor=[0, 1] 范例：CustomCursor=1

{CustomCursor：控制是否使用自定义鼠标指针，当 CustomCursor =1 时，如果安装目录的 Cursors 文件夹里有文件名为'HAND_1.cur'、'HAND_2.cur'、'CROSS.cur'、'MOVE_BEGIN.cur'、'MOVING.cur' 、 'CLKSELECT.cur' 、 'LOCKED.cur' 、 'DRAGHANDLE.cur' 、'HANDDRAWING.cur' 、 'ROTATE1.cur' 、 'ROTATE2.cur' 、 'ROTATE3.cur' 、'ROTATE4.cur'的鼠标指针文件，则载入这些鼠标指针文件，取代系统内定的鼠标指针
}

TrayWndPopup=[0, 1] 范例：TrayWndPopup=1

{TrayWndPopup：控制托盘图标是否弹出信息。新版本的神州易刻的主窗口右上角已有更直观的作业进度信息，TrayWndPopup=0 时，托盘图标仅弹出一些重要的消息，而不会频繁弹出进度信息，使用体验更好一些
}

DefaultFont0=[字体名] 范例：DefaultFont0=宋体

DefaultFont1=[字体名] 范例：DefaultFont1=蒙文图章体

DefaultFont2=[字体名] 范例：DefaultFont2=Arial

{DefaultFont0, 1, 2：点击文字输入窗口有 T、M、W 标记的三个按钮时默认选择的字体
DefaultFont0：T 标记按钮
DefaultFont1：M 标记的按钮
DefaultFont2：W 标记的按钮
}

MultiRunEnable=[0，1] 范例：MultiRunEnable=1

{MultiRunEnable：控制是否允许神州易刻多开
MultiRunEnable =0：不允许神州易刻多开
MultiRunEnable =1：允许神州易刻多开

提示：如果激光雕刻机被多个软件同时控制，多个软件同时向激光雕刻机发送控制命令，那么激光雕刻机就会被多个软件同时指挥，激光雕刻机的动作就要错乱。所以，默认只允许打开一个激光雕刻软件。但是，可能有些用户只需要打印输出或导出图片，并不需要连接激光雕刻机，那么设置 MultiRunEnable =1 即允许神州易刻多开

老版本的神州易刻(LaserDRW)、CorelLASER、WinsealXP 任何时候都只能打开三者之一，要打开另一个软件需要先关闭已打开的雕刻软件。自 2024.06 版本后，允许同时打开神州易刻(LaserDRW)、CorelLASER，遵循谁先开谁占有激光雕刻机的逻辑。
}

EngraverSetting.ini(激光雕刻机控制或配置)

[Scripts]

RunTimes=[大于 0 的整数] 范例：RunTimes=10

{RunTimes：硬件调试窗口复位并多向走空程执行的次数。多向走空程可以很好地检查轨道的机械特性，比如多向走空程前设置首尾自动点射激光，起始时激光自动点射一个点，如果多向走空程很多次后，结束时再自动点射激光，如果起点点射的激光点与结束时点射的激光点完全重合，说明轨道在全幅面上有相对顺滑的性能
}

CutMarginX=[Double] 范例：CutMarginX=1.2

{CutMarginX：切割外形边框时 X 方向的边界，单位为 mm，比如 CutMarginX=1.2，切割外形边框时，X 方向保留 1.2mm 的边界。如果不设置 CutMarginX，那么神州易刻按默认方式切割。比方排版时版面设置增加 2mm，说明左、右、上、下各增加 1mm，按照默认方式切割就是 1mm/2=0.5mm 的边界。但是，也可以设置 CutMarginX 的值自定义 X 方向的边界

}

CutMarginY=[Double] 范例：CutMarginY=0.8

{CutMarginY：切割外形边框时 Y 方向的边界，单位为 mm，比如 CutMarginY=2.1，切割外形边框时，Y 方向保留 2.1mm 的边界。请参考上面 CutMarginX 的说明

}

CutMarginTimes=[>0 的整数] 范例：CutMarginTimes=2

{CutMarginTimes：切割外形边框时，切割几遍。当材料很厚时，或者切割速度快了切不穿，切割速度慢了又容易着火变形，那么可以设置切割次数。雕刻完成后一般切割一次，如果设置 CutMarginTimes =3，那么雕刻完成后切割边框，会重复切割 3 次

提示：如果激光机切割时收口处不完美，那么设置切割 2 次则会使收口更自然完美！

}

ini 文件脚本控制综论

神州易刻 2013.02 版本长达 11 年未曾更新升级，但仍然占据了主流市场，说明了它的基础功能好用、易用、也基本够用，否则它 11 年不更新升级，必然要被淘汰出局。当我计划开发新版本神州易刻时，面对一大堆用户、厂家、网络备案公司等等的诉求，就十分为难，因为易用实际是精简削除一些不太常用的功能，并添加一些自动化处理来达到的。比如神州易刻实际是我方软件 WinsealXP 大砍大削后的产品，大约削除了 WinsealXP 70%以上的功能，才使其符合简单易用的特点。但是，你说要增加这些功能，他说要增加那些功能，一大堆的诉求如果都简单粗暴地捆绑上去，势必就导致神州易刻要背离易字根本，因为用户要了解一大堆他用不着又搞不明白究竟是啥作用的功能，就会头晕脑胀，所以，我们采用了 ini 脚本控制、外部程序协作、第三方 dll 插件挂接的方式，来满足各种不同的需求，以便完美保留并尽可能提升加强其易用性，并使神州易刻的拓展不完全依赖于我个人，也解放了我自己。

ini 脚本控制实质已经比较完全地控制了神州易刻的工作。为什么这样说呢？神州易刻主体功能是排版和输出，以上就介绍了监听图象文件自动排版、监听 yzi 文件套用模板批量排版，启动雕刻时自动导出图象、人工导出图象等等。那么，可以依据这些资料做哪些事情呢?

1、设计神州易刻功能配置中间件（也就是写一个小巧的程序），专门用于配置神州易刻的 ini 脚本控制字，比如

外形切割次数，其控制字 CutMarginTimes 在 EngraverSetting.ini 文件里的[Scripts]字段下面，不太熟悉电脑的用户可能不知道 EngraverSetting.ini 究竟在哪。如果编写一个小巧的实用程序，把 ini 脚本控制字的配置挖掘出来并程序化，那么不懂电脑的用户，就能用这个小程序配置一下他需要的这些隐藏功能；

2、 联网上传与下载：比方某个备案系统里下载的图片，如何自动排版到神州易刻里？雕刻完成后，用户又如何简单快捷上传印模到备案系统？这时可以编写一个外部协作程序，设置 StartSyncApp 控制字为这个外部协作程序的全路径 exe 文件名，由这个外部协作程序自动完成下载、排版、上传等工作。比如该外部协作程序下载图片后，外部协作程序按上面提到的格式化文件名规则格式化图片文件名，并存放到神州易刻的指定监听文件夹 ListenDir 里，神州易刻监听到这个文件，就会按格式化文件名的要求立刻完成排版。可设置雕刻时立刻导出图象(这样不影响继续排版，否则你要等雕刻好了再排版，因为你需要等雕刻好了导出上传用的图片)，雕刻开始时会发消息通知外部协作程序，外部协作程序读取神州易刻导出的图象，并等待神州易刻发送雕刻完成（异常完成、正常完成）的消息，一旦接收到雕刻正常完成的消息，外部协作程序就弹出确认窗口，让用户确认是否立刻上传印模。有些备案系统是自有排版系统，以图片的方式交付给刻制者，而有的备案系统没有排版系统，是以文字的方式交付给刻制者(由刻制者自行排版)。对于以文字交付的方式，使用 yzi 文件来完成自动排版即可。所以，不管备案系统交付给刻制者的是图片还是文字，都能自动排版。

3、 第三方激光雕刻机和 CNC 设备的挂接：因为神州易刻有广泛的用户基础，你可能想制造个性化的激光雕刻机并商业化，但若没有一个广泛用户基础的排版软件，就很难有发展前途，因为神州易刻几乎是不可抗拒地统治了这个行业，你的设备做得再好，若不能用神州易刻可能就是用户不愿意选购的。所以也可以编写一个外部协作程序，因为用户执行人工导出操作时，也会通知外部协作程序，外部协作程序收到用户人工导出图片消息时就立刻读入用户导出的图片，弹出雕刻界面，用户确认后即开始雕刻。当然，也可以挂接 CNC 设备雕刻铜印：由外部协作程序把神州易刻导出的图片转换为 G 代码直接发送给 CNC 设备。我方一直有想法把铜印雕刻机标准化，但一直没时间提到日程之内。因为通用 CNC 雕刻铜印，用户要与数个专业软件打交道，大多数用户不太好学会，厂家售后难度高、时间长。

4、 有些地方刻制一个图章，需要导出两张图片：一张是上传到备案系统，另一张是在电子文件上使用的电子印章(需要白色部分全透明，而图文部分是指定透明度的半透明状态)。神州易刻可设置雕刻时立刻一次性导出这两张图片。

Ini 脚本文件+第三方外部协作程序是控制与协助神州易刻的一种方式，这种方式简单易行，也方便二次开发者控制版权：因为第三方外部协作程序是独立的程序，其如何授权完全是他的开发者自主控制的。上面仅举了 4 个例子说明其应用，实际上如何应用全在于二次开发者与实际需求，比如外部协作程序还可以增加一键签章到 Word 或 WPS 功能，一键发送到 CorelDRAW，还可以增加特效签章功能，比如破边、印泥效果、添加 QR 二维码、边框增加特殊需求的防伪纹线……。

重要提示：外部协作程序不一定仅仅依赖 ini 脚本文件，它也可以调用下面要讲述的动态库(dll)插件服务库 PlugInsService.dll 里封装的功能函数，此时最好把它安装到神州易刻的 Plugins 文件夹里，避免需要设置系统环境变量的麻烦。

动态库(dll)插件的接口服务库

神州易刻的二次开发除了 Ini 脚本文件+第三方外部协作程序的方式外，还有一种与神州易刻无缝集成为一体的方式，这种方式就是给神州易刻编写动态库(dll)插件(后面简称 dll 插件)，该方式提供了更丰富更强大的功能，你可以利用该功能做许多你想做的事情，可以完全接管激光雕刻机的控制，也可以完全无缝接入第三方激光雕刻设备或 CNC 设备。

关键点： dll 插件需要版本号大于 2.0 的 USB-Key 支持，且仅当 USB-Key 插入验证正确时神州易刻才会载入 dll 插件

目的： 我方软件与硬件均是坚持一切皆完全自主研发的技术路线(新版本软件的全矢量的 UI 框架都是我方自主构建的)，没有任何不方便维护的第三方代码或技术，从神州易刻的程序大小(整个安装包仅 2.35M)就可以知道其代码的精练，因为借助第三方代码或技术构建的系统，通常都比较庞大，而且因为技术依赖于第三方，会有一些 bugs 长期存在而无法妥善修正(因为第三方不修正你就没办法)，有些甚至只能长期停留在固定的 Windows 平台，比如长期停留在 Windows XP 系统，其原因就是第三方倒闭不存在了，第三方没了更新，你又自主研发不出来，只能在第三方仅支持老的 Windows 系统的技术上折腾。我方软件长达 11 年没有更新仍然保持旺盛的生命力，没有被人超越过去，其主要原因之一就是想超越我方的都或多或少依赖第三方的技术，整个系统的稳健性很差。这次我们提供 dll 插件扩展功能的目的主要在两方面：其一是我个人很忙，对一些用户的特殊(或小众)需求只能忽视不理，而在没开放二次开发功能时，其他开发者又无法给这类用户开发，新版本神州易刻开放了二次开发接口，便于第三方开发特殊需求的插件(比如与备案系统对接)或做一些功能拓展(比如接入第三方激光雕刻机或铜印雕刻机)；其二、改变神州易刻长期依赖我个人的局面。神州易刻长达 11 年没有升级，主要原因就是我个人还有其他方面的业务，实在照顾不到，现在开放了接口，其他开发者也可以开发，并因此获得一定的收益，有利于提升神州易刻的总体经济价值，比方说在神州易刻上挂接个直接雕刻铜章的插件，那么至少就是以百万计算的经济效益。也就是说，新版本的神州易刻也可看作是一个小型的创业平台。

提示： 如果你想研发 CO2 激光雕刻机控制板，最好使用完全隔离的方案，而不要学我们这种非隔离方案，因为我方的软硬件都经过了特殊的算法处理，但这些特定算法仅适用于我方控制板，并不能照顾到第三方开发的控制板。通讯方式不建议选择 USB 转串口(或 CDC 串口)，因为这类 USB 虚拟串口通讯是远远抗不住高压 CO2 激光电源工作时的强干扰，会导致软件死锁(无法关闭)、崩溃(闪退)、计算机蓝屏(Windows 内核关键区被修改)等致命问题，也就是说这类主板通常很难雕刻完一个小幅面作品(刻枚图章都不见得能顺利完成)，大幅面作业是想都不用想。正因为这个原因，一些开源的且流行的 CNC 控制板，至今无法用到 CO2 激光机(半导体激光机基本都是这类开源控制板)，而很多开发者就是在开源的 CNC 控制板(USB 虚拟串口通讯)上改造，试图用在有超强干扰的 CO2 激光机，而且还想学我方不隔离，所以基本都是失败告终。为了增强抗干扰，他们都是加电容、加电感、加共模电感，但实际用处并不大，而我方主板十分简洁，根本没有什么大电容、大电感，因为我们知道无用，只是徒增成本。如果一定要使用 USB 虚拟串口通讯，可以考虑这样的执行流程：先下载数据到控制板的存储器，待数据传送完成再启动雕刻，并立刻在 PC 端关闭 CreateFile 打开的 IO 端口，也就是完全脱机雕刻，数据下载过程高压激光电源处于不工作状态。但若学我们一边下载数据一边工作，大概率这种控制板都要以失败沮丧告终的，就我所知道的，这些年来失败的不少于十个，这也是神州易刻 11 年未曾升级仍能掌控市场的重要原因之一。坦白地说：我这个做工业控制产品为主的人都没搞定 USB 虚拟串口用在 CO2 激光雕刻机上，因为我也试过，我也搞不定它！脱机作业的目的不只是为了方便，另一个重要原因是可以有效避开高压 CO2 激光电源的强干扰，那些严谨隔离的数千元的控制板，使用的也是 USB 虚拟串口通讯，大概率也是不能象我方主板这样一边雕刻一边下载数据的，但脱机正好有效避开。我们设计控制板的目的是为了推广我们的软件，所以有人能设计更好的控制板接口到我们的软件，这是我们十分欢迎的。

(1) 神州易刻的 dll 插件安装到什么地方？如何获取 dll 插件的安装位置?

神州易刻安装时会在系统注册表里写入它的安装路径，所以插件开发者只要读取注册表里的 HKEY_LOCAL_MACHINE\SOFTWARE\Lihuiyu Studio Labs.\ LaserDRW\Install\InstallPath 的值，即可定位 LaserDRW.exe 文件的位置，我们假如读取到的数据是 C:\Program Files\Lihuiyu Studio

Labs\LaserDRW\LaserDRW.exe，那么 dll 插件安装的位置为 C:\Program Files\Lihuiyu Studio Labs\LaserDRW\ Plugins\，后面简称 Plugins 文件夹。编写好的插件应制作一个安装包，一键安装，因为人工拷贝文件的方式，会导致很多麻烦。

(2) 插件服务库 PlugInsService.dll(在 Plugins 文件夹里)提供基础功能函数，它封装了一些插件接口服务函数

1、HWND WINAPI GetAppMainWnd(VOID);

{GetAppMainWnd：获取神州易刻主程序窗口的句柄。什么时候需要神州易刻主窗口的句柄呢？举个例子来说，当你编写的 dll 插件有一个窗口要在顶层模态显示，模态谁？这时就要获取神州易刻主窗口的句柄，并接管其输入输出。如果不模态显示，那么你的插件的窗口可能就会被神州易刻主窗口遮盖住，除非你从系统级 Z 序置顶它

}

2、HWND WINAPI GetAppRightBarWnd(VOID);

{GetAppRightBarWnd：获取神州易刻主程序右侧属性面板的窗口句柄。什么作用？你可以通过替换其窗口函数进行子类化，在上面画点你需要的东西，或者在不影响使用的空白区(比如底部)添加一些按钮执行你需要的功能。但我们并不建议改造右侧属性面板，即便想改造也不可影响用户的使用体验，尤其不可动不动就重画整个客户区，这样会造成严重的闪烁！一些初级开发者不知道如何裁剪 DC，动不动就直接重画整个客户区，闪烁得非常厉害，严重影响用户的使用体验。神州易刻任何时候都是平静如水的！

}

3、typedef VOID(CALLBACK *lpPaintProc)(HDC DC); //定义一个回调函数

LONG SetRightBarPaintProc(lpPaintProc PaintRightBar);

{SetRightBarPaintProc：设置自绘右侧属性面板的函数，该函数的声明就是上面 typedef 所声明的。你可以编写一个符合 typedef 语句定义的函数，然后用 SetRightBarPaintProc 函数把你编写的函数的地址传递给神州易刻，那么神州易刻就会执行你编写的函数，照你的想法额外绘制右侧属性面板。这时候，你可能就要用上面的 GetAppRightBarWnd，因为你需要知道属性面板的大小以及确定绘制在什么位置。SetRightBarPaintProc 返回值恒为 0

}

4、LONG WINAPI EnableActionsExt(DWORD dwEnables);

{EnableActionsExt：使能神州易刻雕刻机菜单的 Actions。雕刻机菜单下有雕刻、导入刀路文件、……、雕刻机初始化设置共 9 个 Actions，我们可使能或禁用它们。dwEnables 的低 9 位 bit0 - bit8，分别设置对应 Actions 的状态，为 1 表示使能，为 0 表示禁用。另外，dwEnables 的 bit16 设置启动/暂停状态，bit17 设置联机/未联机状态。神州易刻启动时若未联机，工具栏并不会显示雕刻机相关的按钮，所以应设置联机状态触发工具栏创建相关的按钮

EnableActionsExt 有什么意义？便于无缝接入第三方激光雕刻机，也就是说你希望用户点击雕刻机

相关的按钮(还有菜单)，显示的不是我方雕刻机的界面，而是你自己的雕刻机的界面，你就要完全接管这些 Actions，后面会提到

}

5、DWORD WINAPI GetDeviceInfo(DWORD dwInfoIndex);

{GetDeviceInfo：获取设备(激光雕刻机)的状态信息

dwInfoIndex = 0：获取激光雕刻机联机状态；返回值非 0 表示激光雕刻机已联机。已联机不一定可使用，联机且注册成功才可以使用

dwInfoIndex = 1：获取编译器状态；返回值非 0 表示编译器正在编译数据

dwInfoIndex = 2：获取是否在向雕刻机发送数据(雕刻数据或命令数据)：返回值非 0 表示正在向雕刻机发送数据

dwInfoIndex = 3：获取是否正在执行雕刻任务。返回值非 0 表示正在执行雕刻任务

dwInfoIndex = 4：获取雕刻机是否处于雕刻暂停状态：正在雕刻但暂停了雕刻，注意不是停止了雕刻。返回值非 0 表示处于雕刻暂停状态。

dwInfoIndex = 5：获取是否正在发送命令给雕刻机。返回值非 0 表示正在发送命令。

dwInfoIndex = 6：获取雕刻作业队列里的作业数量。返回值为雕刻作业队列里的作业数量。神州易刻是可以多个作业送入作业队列的(依进入作业队列的先后次序进行雕刻，雕刻完一个会提示)，也可以是多个作业合并为一个作业，比如把雕刻与切割合并为一个作业，作业合并后会一次性连续作业，不会提示

dwInfoIndex = 7：获取雕刻头的逻辑位置。返回值是一个 Double 型指针，指向一个数组 Double[2]的首地址。Double[0]是 X 轴的逻辑位置，Double[1]是 Y 轴的逻辑位置，单位是 mm。所有指针(也就是数据存放地址)实际都是 ULONG(DWORD)。不要迷惑 GetDeviceInfo 的返回类型不是 DWORD 吗，怎么变成了 Double 指针，是不是搞错了？没有的错！

提示：X 轴与 Y 轴是相对的，不一定是具体的机器规定的 X、Y，也就是说，对于一台具体的机器，你获取的 X 数据与 Y 数据有可能恰好相反

dwInfoIndex = 8：获取激光雕刻机的注册状态。返回值非 0 表示注册成功，可以使用

dwInfoIndex = 9：获取激光雕刻机主板的硬件特征综合信息，比如芯片特征码、固件特征码、出厂流水号、出厂日期等特殊信息。该功能对二次开发无用

dwInfoIndex = 11：获取是否支持软件控制激光能量特性。返回值非 0 表示支持

提示：仅固件版本号大于或等于 2024.01.18g 的 M3 主板才支持软件控制激光

dwInfoIndex = 12：获取主板是否支持硬件配置功能。返回值非 0 表示支持

提示：M3 系列全支持硬件配置功能，但不一定支持软件控制激光能量

dwInfoIndex = 20：获取当前进度(编译器编译进度，激光雕刻机雕刻进度)。

返回值 HIWORD(高 8 位)=（1，2）[1=雕刻进度、2=编译进度]

返回值 LOWORD(低 8 位)为当前进度百分数。

比如返回值 0x00010032 表示雕刻(0x0001)进度 50%(0x0032)。

dwInfoIndex = 107：获取激光雕刻机是否禁用/使能了复位功能。返回值非 0 表明已使能

以上列举了 GetDeviceInfo 函数的一些参数，这些参数可能二次开发用得着。其他用不上的参数就不一一说明了。

}

6、DWORD WINAPI SetDeviceCommand(DWORD dwCommand, DWORD dwData);

{SetDeviceCommand：控制设备(激光雕刻机)。这个函数只有两个参数，但它是一个非常复杂的函数，如果不了解指针的程序员，可能会看得晕头转向的，我先做点说明！

常识：指针即地址(编号)，所有指针都是整型数据，所有指针都可以强制转换为 ULONG、DWORD、LONG 型数据作为函数参数，只不过该整型数据是地址(编号)而已。利用指针特性，所有函数的参数实质都可以简化为仅 1 个参数(一个指针指向参数结构体(或参数链表))，多个参数仅仅是为了易于理解、可读性更好而已！如果不理解指针，那就不太容易理解为什么仅一个 SetDeviceCommand 函数且只有两个参数，就能完成雕刻机的全部控制

dwCommand：命令索引号

dwData：对应命令的数据(可能是直接数据，也可能是一系列数据存放的首地址(指针))

dwCommand = 0， dwData =指针(*VOID)：该地址依次存放有主板的逻辑验证码。该功能二次开发用不上，因为主板逻辑验证码并非输入的明码，而是经过加密运算后的复杂码

dwCommand = 1， dwData =直接数据(HWND)：传入一个窗口句柄，雕刻系统会发送状态信息给该窗口。该功能二次开发用不上，因为神州易刻主窗口已经接管了相关消息的处理

dwCommand = 2， dwData =直接数据(取值 0 - 5)：设置雕刻线程的优先级。一般而言线程的优先级是相对本软件自身的。大多数时候保持默认不用设置

dwCommand = 3， dwData =指针(*ULONG[2])：设置坐标系统；ULONG[0]取值[0\1\2\3]=[左上原点、右上原点、左下原点、右下原点]，ULONG[1]取值[0\1]=[激光头水平运动\激光头垂直运动]。二次开发不建议使用非标坐标系，因为存在一系列的变换，十分烦琐，所以保持默认的左上原点且激光头水平运动的坐标系即可

dwCommand = 4， dwData =指针(*Double)：设置脉冲当量值(单位：纳米(nm)/脉冲)。该脉冲当量实际是逻辑 X 轴的脉冲当量。逻辑 Y 轴的脉冲当量不用传入。是不是只能设置一个轴的脉冲当量？不是，逻辑 Y 轴的脉冲当量实际是主程序设置的，因为只要以逻辑 X 轴为基准，缩放逻辑 Y 的数据即可。重提一次：X、Y 是相对的，如果你开发发现 X 好象是你的图的 Y，那么你就当你的图的 Y 是 X 即可，X、Y 只是个符号而已

dwCommand = 5， dwData =指针(*LONG[2])：设置 X、Y 的逆程补偿数据，可能是正数也可能是负数，但一般在(- 4 - +4)之间。LONG[0]是 X 的逆程补偿数据，LONG[1]是 Y 的逆程补偿数据。逆程补偿是以一个方向为基准，补偿逆程时因为机械误差、激光系统延迟的影响，可让雕刻效果更优，雕刻精度更优

dwCommand = 6， dwData =直接数据(取值 0 - 2)：设置当前作业是雕刻(0)、切割(1)、打标(2)。一般只要设置雕刻(0)和切割(1)，打标(2)是矢量填充模式，是为雕刻铜印准备的，只是一直没空研发而已。提示：神州易刻支持位图雕刻与位图轮廓切割

dwCommand = 7， dwData =直接数据(取值 0 - 3)：设置版面方向，不旋转(0)，左转 90 度(1)，右转 90 度(2)，旋转 180 度(3)。提示：后面传入的雕刻图象或数据，不用旋转处理后再传入，系统会自动处理，二次开发者只要设置一下即可

dwCommand = 8， dwData =指针(*Double[6])：设置各种速度和激光能量。
Double[0]=雕刻速度(单位 mm/s)，Double[1]=雕刻激光能量(单位%)
Double[2]=切割速度(单位 mm/s)，Double[3]=切割激光能量(单位%)
Double[4]=预览速度(单位 mm/s)，Double[5]=预览激光能量(单位%)
分层雕刻时，每层数据发送到编译器之前，都要设置一下该层对应的工作速度和激光能量

dwCommand = 9， dwData =直接数据(1-8)：设置象素步长。如果激光雕刻机的精度是 1000dpi，象素步长是 n，那么我们只需要送给编译器 1000/n dpi 的图片即可。象素步长的意义在于：其一提高雕刻效率，使激光机能加工更大的幅面；其二、防止过于密集雕刻引起着火烧焦的问题

dwCommand = 10， dwData =直接数据(0/1)：设置镜像雕刻，镜像(1)，否则(0)。对于不同坐标系统，设置了镜像是否真需要镜像？如果不做任何处理，左上坐标原点的设备刻出来的是正字，则右上坐标原点的设备刻出来的恰好是反字。因此，为了刻章(都是反字)而诞生了右上坐标的激光机，因为不用处理图片直接刻就是反字。神州易刻里任何时候都不会使用旋转图片的方式处理雕刻数据，因为旋转一张图片至少要在内存里占用两张图片的存储空间，若图片很大不但耗时而且可能导致资源耗尽的问题，也就是说本来该计算机的内存可满足 500mmx500mm 的雕刻幅面，但因为要旋转图片，导致可处理的幅面减半，而且处理效率低下，使用体验差。神州易刻里二次开发时传递给编译器的雕刻图片，不用做旋转、镜像处理，编译器会用改变扫描位置和方向、不多耗任何资源的算法自行高速处理，但一定要设置正确的坐标原点，否则结果就是错误的。我们建议优先制造左上原点的设备，直观好用不别扭，因为神州易刻镜像和旋转版面，与不镜像且不旋转版面，其处理速度基本是一样的。
提示：预览图旋转镜像是给用户看的，但雕刻时内部是以改变扫描起始位置和方向的方法处理

dwCommand = 11， dwData =直接数据(0/1)：设置阴刻阳刻，阳刻(0)，阴刻(1)。

dwCommand = 13， dwData =直接数据(0/1)：设置单向雕刻，双向刻(0)，单向刻(1)。

dwCommand = 16， dwData =直接数据(0：忽略该参数)：启动雕刻(或暂停后重启雕刻)。
提示：编译器编译好数据后，会送入雕刻队列，但并不会启动雕刻，因为本系统是可以无限增加雕刻作业，也可以无数个雕刻作业合并的。比方分层雕刻切割，处理好每层数据后，可能需要合并每层的数据，整体作为一个雕刻作业。所以，如果要启动雕刻，可以发送 16 号命令，但更方便的是下面的 37 号命令

dwCommand = 17， dwData =直接数据(0：忽略该参数)：暂停雕刻(仅暂停，可恢复)

dwCommand = 18， dwData =直接数据(0：忽略该参数)：停止雕刻(不可恢复)

dwCommand = 19， dwData =指针(*Double[2])：设置逻辑坐标(即激光头定位到什么地方)。
Double[0]=X 坐标(mm)， Double[1]=Y 坐标(mm)

dwCommand = 20， dwData =直接数据(>=1 的整数)：设置重复雕刻次数

dwCommand = 21， dwData =指针(*一个复杂结构体)：向任务队列添加外形切割线。该命令号的 dwData 太复杂了，本身是一个结构体的指针，而结构体中还有指针，可能不太好说明白。我们建议

二次开发时要自动切边框或预览位置，可以用这样的方法：创建一张白底色的单色位图，位图上用笔宽为 1 象素的黑笔画好要切割的边框，设置为切割模式，然后直接把该位图送给编译器，编译器有一个非常高效的求取轮廓算法。因为 1000dpi 的 1 米 x1 米幅面的单色位图在内存里也只 185M 左右，并不算很吃内存。这个位图你只要正常画即可，不用处理旋转、镜相等算法，画好直接交给编译器即可。注意要以设备的分辨率创建该位图，不要考虑象素步长，因为象素步长仅对雕刻有效，也就是说切割强制象素步长为 1

dwCommand = 23， dwData =指针(*一个复杂结构体数组)：增加一组外形切割线。比上面的 21 号命令还要复杂。同样，二次开发不妨象上面一样，只是这张位图上画了很多个边框而已

dwCommand = 24， dwData =直接数据(HBITMAP)：把一张位图(建议是单色位图)的句柄交给编译器，这张位图是不需要处理旋转和镜相的(即使设置了镜相或旋转)。编译器提取特征数据后，并不会删除该位图，开发者不需要该位图时，需要自己删除它
提示：正式把数据交给编译器之前，要正确设置好其他参数(比如控制板型号、坐标系统、版面旋转、镜相、雕刻还是切割、工作速度、激光能量等等)，因为编译器获取位图的特征数据后立刻按照已经设置好的参数进行编译

dwCommand = 28， dwData =指针(*char)：导入刀路文件。dwData 参数为刀路文件名第一个字符的首地址，文件名应以 NULL 结束

dwCommand = 29， dwData =指针(*char)：雕刻到文件。dwData 参数为保存刀路文件名第一个字符的首地址，文件名应以 NULL 结束

dwCommand = 29， dwData =直接数据(0 - 6)：设置控制板型号。A(0)、B(1)、B1(2)、M(3)、M1(4)、M2(5)、M3(6)

dwCommand = 33， dwData =直接数据(0：忽略该参数)：复位激光雕刻机

dwCommand = 34， dwData =直接数据(0：忽略该参数)：激光雕刻机重新定位

dwCommand = 35， dwData =直接数据(0：忽略该参数)：释放马达

dwCommand = 36， dwData =直接数据(0/1)：合并雕刻作业。不合并(0)，合并(1)。比方说我们希望雕刻后立即切割，那么我们就要预先设置为合并雕刻作业。然后把待雕刻位图交给编译器，接着把要切割的位图交给编译器(要记得使用 6 号命令把编译器切换到切割)，编译器就会把雕刻和切割合并为一个任务

dwCommand = 37， dwData =直接数据(0/1)：编译完成后是否立即启动。不启动(0)，立即启动(1)如果只有一个任务，可以设置为立即启动。如果有多个任务，交给编译器最后一个任务前，设置为立即启动。否则，你要用 GetDeviceInfo 不断查询编译器是否编译结束，以便及时启动雕刻

dwCommand = 38， dwData =直接数据(0：忽略该参数)：清空雕刻作业队列

dwCommand = 46， dwData =指针(*Double)：邻线合并阈值(mm)。编译切割数据时，编译器会依

据该阈值合并端点间隔小于该值的线条

dwCommand = 47， dwData =直接数据(0 - 5)：优化切割数据。一般设置为 4(先切割内部)即可，其他优化模式神州易刻可不必选用

dwCommand =101， dwData =直接数据(0/1)：启用中断传输模式，仅 M3 以上的主板支持。通讯(A 向 B 发送数据)有两种基本的握手协议： 其一、A 定时去问 B 能不能接收数据(B 等待 A 来问，可能要问几次)；其二、B 可以接收数据时，马上通知 A 发送数据(A 等待 B 通知，一次直接送达)。这两种握手方式即定时轮询、中断传输，前者效率低而后者效率高。M3 之前的主板一直使用智能定时轮询的方式，为什么呢？因为 CO2 激光机的高压激光电源的强干扰，可能导致 B 通知 A 的信号丢失(A 没收到通知)，其后果是致使 A 一直在死死地等 B 的通知，所以采用亲自上门去问的方式明显更安全可靠。但有的开发者不知所以然，还特地攻击了一下我这专门研发的通讯。M3 主板则可以启用中断传输方案，加速数据传输，但它的后盾依然是智能定时轮询。实际上很多开发者的主板主要就是死在通讯这里！

dwCommand = 104， dwData =直接数据(0：必须是 0)：点射激光

激光雕刻机完全由 SetDeviceCommand 函数控制，我们仅整理了一些二次开发用得着的命令号，其他命令号可能作用不大，或者其 dwData 参数解释清楚过于繁难，只好略过

}

7、VOID WINAPI PageSwitchTo(LONG PageIndex);

{PageSwitchTo：切换到设计版面或组版版面。

PageIndex =0：切换到设计版面

PageIndex =1：切换到组版版面

}

8、LONG WINAPI GetPageShapesCount(LONG PageIndex);

{GetPageShapesCount：获取版面上的图形数量(返回值即数量)

PageIndex =0：获取设计版面上的图形数量

PageIndex =1：获取组版版面上的图形数量

有什么作用？比如需要导出图形前，开发者可先用此函数查询一下相应的版面上是否有图形

}

9、LONG WINAPI ExportImage(LPWSTR lpSavePath,

DWORD dwColor, DWORD dwFlags);

{ExportImage：导出图象。返回值为 0 失败，非 0 成功

lpSavePath：导出图象的保存路径

注意：只提供路径即可，不需要文件名，文件名是依据导出时间自动生成的

dwColor： 图象渲染什么颜色

dwFlags：Bit0 – Bit11： 导出图象的 dpi

Bit12 – Bit15：导出图象的格式

0x0000-0x6000 对应：bmp、png、jpg、gif、emf、tiff、pcx

Bit16 – Bit23：png 图象的 alpha 通道值
0x000000-0xff0000 对应 0-255
Bit24 : =1 同时输出带 alpha 通道的 png，否则 0
Bit25 : =1 修剪掉图片四边多余的空白，否则 0
Bit28 : =1 导出单色图象，否则 0
Bit29 : =1 导出的图象送到剪贴板，否则 0(文件名在剪贴板)
Bit30 : =1 导出组版页面的图象， =0 导出设计版面的图象

提示：前面 ini 脚本控制部分，说了两种导出方式：启动雕刻自动导出、用户人工导出。使用此函数，则想导出就导出，不受限制。ExportImage 导出的是图象文件，还有一个 GetHBitmap 函数也能导出图象，且 GetHBitmap 导出的图象是内存句柄(即 HBITMAP)，详见后文说明
}

10、LONG WINAPI DesignSaveToFile(LPWSTR lpFileName, LONG PageIndex);
{DesignSaveToFile：把设计保存为文件。返回值为 0 失败，非 0 成功
lpFileName：保存文件名。设计版面扩展名应为.lyz，组版版面扩展名应为.pyz
PageIndex：0：保存设计版面；1：保存组版版面
}

11、LONG WINAPI GetShapesOutlinePointsCount(VOID);
{GetShapesOutlinePointsCount：获取组版页面各个图形的外形切割线需要的 Point 数量，以便后续分配内存；该函数主要用于接口第三方激光雕刻机
}

12、HBITMAP WINAPI GetHBitmap(Double nmPerPixel, Double nmPlusEqu,
Double mmCutMargin, POINT *lpBuffer,
DWORD dwFlags) ;
{GetHBitmap：获取匹配激光雕刻机精度的位图。该函数主要用于接口第三方激光雕刻机。当然也可以是其他应用
nmPerPixel：需要的图象精度，非 dpi，而是一个象素的直径是多少纳米(nm)，这样才能生成精准的图象，避免雕刻时进行缩放处理丢失精度
nmPlusEqu：设备的脉冲当量(单位是纳米(nm)/脉冲)。激光雕刻机雕刻用的图象不一定是设备自身的 dpi，比如 2000dpi 的雕刻机，雕刻用的图象可能只需要 500dpi(象素步长 4)，但切割一般用 2000dpi 最好。nmPlusEqu 用于计算外框切割数据，比如 2000dpi 的雕刻机计算的数据就是适配 2000dpi 的
mmCutMargin：外框切割边界(单位：mm)。也就是切割线离雕刻好的图形外缘多远切割
lpBuffer：该参数是一个 Point 数组，用于存放外框切割数据的关键点。该参数需要的内存由上面的 GetShapesOutlinePointsCount 预先计算好，并分配好内存
lpBuffer 存储的数据：如果 lpBuffer[n].Y=2147483647(MAXINT)说明 lpBuffer[n]是一组外形边框数据的起始数据，而 lpBuffer[n].X 存储的是外形边框线类型：
lpBuffer[n].X = 0(矩形：2 Points) //矩形的左上、右下
lpBuffer[n].X = 1(椭圆：2 Points) //椭圆的绑定矩形
lpBuffer[n].X = 2(菱形：4 Points)
lpBuffer[n].X = 3(下三角：3 Points)

lpBuffer[n].X = 4(上三角：3 Points)

lpBuffer[n].X = 5(左三角：3 Points)

lpBuffer[n].X = 6(右三角：3 Points)

而从 lpBuffer[n+1]开始存储这个外形边框的关键数据。例如 lpBuffer[n].X = 3，说明该外框是下三角形，那么 lpBuffer[n+1]、lpBuffer[n+2]、lpBuffer[n+3]中存储了这个三角形的 3 个顶点。而 lpBuffer[n+4]则是下一个外形边框的起始数据

lpBuffer 中存储的是外形边框线的原始数据(可指定是否镜相)，如果雕刻图形时做了版面旋转等处理，切割时也应对外形边框线做同样的版面旋转处理

dwFlags：Bit0：=0 必须恒为 0

Bit1：=0 不裁剪图象； =1 裁剪图象

Bit2：=0 必须恒为 0

Bit3：=0 不要遮罩； =1 包含阳刻遮罩

遮罩即不需雕刻的空白区遮盖住，避免不必要的激光损耗

Bit4：=0 必须恒为 0

Bit5：=0 仅需要图象； =1 同时获取外形切割数据(外形切割数据填充到 lpBuffer)

仅组版版面有外形切割数据

Bit6：=0 不处理； =1 把外形切割数据水平镜相

Bit7：=0 必须恒为 0

Bit9 Bit8：=01 获取设计版面的图象； =10 获取组版版面的图象

Bit31… Bit10：必须恒为 0

提示：GetHBitmap 这个函数主要为接口其他雕刻机用，但它可以灵活应用在许多场合。首先，该函数不会在计算机上产生任何文件，返回值是 HBITMAP 句柄(实际是指针)。其次，该函数导出组版页面是整体导出为一张图片。但是，GetHBitmap 导出的是黑白的位图，有了黑白的位图，想变成什么颜色都很容易，因为你只要用光栅运算 Paint(OR 运算)一下即可。Windows 系统里所有图片显示出来，都是要转换到位图，所以有了位图，转换成其他格式自然也是不难的。所以，如果你不是接口激光雕刻机，而是希望导出图片不产生任何可见文件的话，就要用 GetHBitmap 了，这时设置 dwFlags 的 Bit5=0(仅需要图象，不需要外框切割数据)，nmPerPixel = 25400000 / (导出图象的 dpi)，nmPlusEqu = nmPerPixel， mmCutMargin=0.0， lpBuffer = NULL，dwFlags 的其他 bit 按需求设置。但是要注意的是，外部协作程序或许不行(没测试过)，因为存在跨进程了，如确有跨进程需求，后续版本考虑增加跨进程获取内存位图的函数

}

13、LONG WINAPI CreateDesignFromFile(LPWSTR lpFileName);

{CreateDesignFromFile：直接命令神州易刻把 lpFileName 指定的文件排版到页面上，该文件类型必须是 bmp png jpg jpeg gif emf wmf tiff pcx yzi

请回头查阅 ini 文件脚本控制部分。返回值：成功返回非 0，失败返回 0

应用举例：比如备案系统创建了一个 yzi 文件，这个 yzi 文件如何交给神州易刻去排版呢？可以把 yzi 文件送到神州易刻的监听目录，也可以直接使用 CreateDesignFromFile 函数

}

14、LONG WINAPI GetApplicationIniFile(LPWSTR lpBuffer, DWORD dwType);

{GetApplicationIniFile：获取神州易刻配置文件(RunOptions.ini 和 EngraverSetting.ini)全路径文件名

lpBuffer：获取的文件名存放的缓冲区。一般而言，应预先给 lpBuffer 分配 MAX_PATH(260)个 WCHAR 的空间

dwType：dwType =0，获取 RunOptions.ini 全路径文件名

dwType =0，获取 EngraverSetting.ini 全路径文件名

提示：如果插件需要修改配置文件，或插件是专门用于修改这两个配置文件的，那么就可以调用 GetApplicationIniFile 获取到 RunOptions.ini 和 EngraverSetting.ini 的全路径文件名

}

PlugInsService.dll 插件服务库只有 14.5k，但要把其资料写清楚，也很耗时。所以，开放二次接口对我们来说也不容易，因为写资料太耗时间了！所谓插件服务库就是服务于二次开发者的，也就是二次开发者可利用 PlugInsService.dll 提供的功能函数，去完成自己的插件。也就是一组 API。

动态库(dll)插件开发

dll 插件是比较常见的技术，作为一个开发者，若你不了解这个技术，我这里的资料或许还能让你学习一下 dll 插件的实现技术，窥探下很多资料里语焉不详的奥秘。编写神州易刻的 dll 插件，至少需要在 dll 插件内编写两个函数：

GetPluginsNameAndID：插件的功能定义与功能描述函数

PluginsExecute：插件的功能执行函数

也就是说插件开发者要按照下面定义的原样编写这两个函数，不得随意更改函数声明，因为神州易刻要依赖 GetPluginsNameAndID 识别这个帮忙的人，是来帮什么忙的，神州易刻还要调用帮忙者的 PluginsExecute 去完成具体的任务，实际也就是 dll 插件来帮什么忙，如何帮忙！

开放三类插件功能 ID(神州易刻共有十几个插件 ID，一些属于我方自用的，一些属于未来备用的)

```
#define   TRANSLATE_ID     0x00000008    //翻译插件的功能 ID：插入到文字输入窗口的翻译菜单
#define   EXTACTIONS_ID    0x00000009    //激光雕刻机菜单挂接外部 Actions 的功能 ID
#define   DIALOGBOX_ID     0x0000000A    //对话框(窗口)式插件的功能 ID，当然不一定非要窗口
```

1. 插件的功能描述函数(是你的 dll 插件里必须编写的函数，神州易刻是其调用者)

```
LONG   WINAPI   GetPluginsNameAndID(LPWSTR   lpBuffer);
```

{GetPluginsNameAndID：复制你的插件的名称到 lpBuffer，不要超过 63 个字符，插件名称会添加到增强插件菜单或文字输入窗口的翻译菜单！

该函数返回值必须是 TRANSLATE_ID 、EXTACTIONS_ID 、DIALOGBOX_ID 三者之一，你的插件属于哪种，就返回哪种 ID，不可以随便返回

提示：不管什么功能的插件，GetPluginsNameAndID 都是这个样式！但插件功能执行函数的名称都是 PluginsExecute，但参数序列却不尽相同

}

2. 插件的功能执行函数(是你的 dll 插件里必须编写的函数，神州易刻是其调用者)

(1) TRANSLATE_ID //0x08：翻译功能插件

```
LONG   WINAPI   PluginsExecute(HWND   wnd,          //(in) 请求翻译的窗口句柄
                                LPWSTR   lpSource,  //(in) 待翻译的文本
                                LPWSTR   lpDest,    //(out) 翻译后的文本
```

```
                                    LPWSTR   lpFontName, //(out) 翻译后的文本需用什么字体显示
                                    LONG     Count          // (in) lpDest 缓冲区的长度
                                    );
```

{PluginsExecute：返回值为翻译后的文本长度，也即 lpDest 的长度

因为 lpSource 翻译成 lpDest，长度不一定一样，所以第一次调用时，lpDest 无法分配内存，所以该函数应这样设计：若传入的 lpDest 为 NULL，说明是要求该函数返回翻译后的文本长度，神州易刻会依据这个长度为 lpDest 分配好内存并第二次调用 PluginsExecute，这时把翻译结果复制到 lpDest 即完成了翻译，当然同时也要把可显示翻译结果的字体名复制到 lpFontName，lpFontName 最多是 31 个字符(WCHAR)，这是 Windows 系统规定的。

提示：Windows API 很多都是这样的 2 次调用逻辑！但不少人调用这类 API 时，不知其内幕技术，喜欢用一次调用方式，分配一个自认为足够大的内存，这是内存泄漏的一种来源，因为你认为足够大，但假如用户无意中粘贴了大量字符呢？可能有人疑惑：你这么调用两次，那不是执行效率低了一倍？当然不会：比方第一次要计算翻译结果的长度，不翻译好如何计算？第一次就翻译好了，但发现 lpDest 是 NULL，所以暂存翻译结果，第二次调用只是迅速把暂存的翻译结果复制到已分配好内存的 lpDest 而已！这是确保不会发生内存泄漏的调用方法。Count 参数是告知 lpDest 最多可容纳多少个 WCHAR，避免调用者少分配了内存你却复制了超量信息而造成内存泄漏。Windows API 有很多是这样的 2 次调用逻辑！

注意：神州易刻第一次调用是这样的：PluginsExecute(Wnd, lpSource, NULL, NULL, 0)，这第一次相当于：请给我备货（翻译好），备好货后告诉我要带多少钱去提货；第二次调用是直接带上正好合适的钱去提已经备好的货。OK?!

}

(2) EXTACTIONS_ID　　//0x09：激光雕刻机菜单挂接外部 Actions 的功能 ID

```
LONG   WINAPI   PluginsExecute(HWND   wnd,              //神州易刻主窗体句柄
                               LONG   ActionIndex,     //Action 索引
                               LPWSTR   lpActionName   //Action 菜单名
                              );
```

{PluginsExecute：返回值不重要，可以恒为 0。

Wnd　　：神州易刻主窗体的句柄，主要方便模态化显示你插件里的窗口，否则你想模态化显示却不知相对谁去模态。

ActionIndex：这是告诉你用户要执行哪个 Action。雕刻机菜单下有雕刻、导入刀路文件、……、雕刻机初始化设置共 9 个 Actions，其 ActionIndex 依次是 0 – 8。如下：

```
switch(ActionIndex)
{   case 0:
    {//do something   雕刻
      break;
    }

    case 1:
    {//do something   导入刀路文件
      break;
    }
```

```
            case 2:
            {//do something   清除任务
              break;
            }

            ......
            default:
              break;
          }
```

lpActionName：指向 Action 的 Caption(菜单文字)的指针，调试时你可以检查 lpActionName 的内容核对究竟是在执行哪个 Action，它与 ActionIndex 的对应关系，没什么具体的作用

提示：接口第三方激光雕刻机有两种方法，一种是编写 EXTACTIONS_ID 插件。这种方法直接把神州易刻所有与激光雕刻机相关的菜单、按钮，直接绑定到第三方激光雕刻机，好象是原生的一样。另一种方法是编写 DIALOGBOX_ID 插件，该方法是在增强插件菜单下添加了一条菜单，用户用第三方激光雕刻机雕刻，要去增强插件菜单下执行

因为你的 EXTACTIONS_ID 插件已经接管了神州易刻所有与激光雕刻机相关的 Actions，你也需要控制这些 Actions 的状态，如何控制？调用插件服务库 PlugInsService.dll 里的 EnableActionsExt 函数即可控制这一组 Actions 的状态

}

(3) DIALOGBOX_ID //0x0A：对话框(窗口)式插件的功能 ID，当然不一定非要窗口

```
BOOL WINAPI PluginsExecute(HWND   wnd,          //神州易刻主窗体句柄
                           LONG   PageIndex,    //0：设计版面 1：组版版面
                           LONG   ShapesCount   //当前版面的图形个数
                           );
```

{PluginsExecute：wnd(神州易刻主窗体句柄)，PageIndex(当前是在什么版面)，ShapesCount(当前版面的图形数量)，都是神州易刻传入的数据。插件安装后会在增强插件菜单下添加该插件的专属菜单，菜单的文字就是你的插件里的 GetPluginsNameAndID 函数返回给神州易刻的插件名称

这是一个多功能插件，可以实现各种各样的功能，比如接口第三方激光雕刻机、自主控制神州易刻原生的激光雕刻机、接口 CNC 设备、接口激光打标机、对接各类备案系统、自动排版、发送电子印章到 Word/WPS、二次处理印章(比如印章添加特殊纹线、添加 QR 二维码等)、图象处理(比如雕刻照片，处理照片后利用自动排版功能让神州易刻排版后雕刻)……。

可能有的人迷惑了，就这个函数能干什么呀？因为该插件只是呼出你的窗口，然后你窗口上可以设计菜单、按钮，你可以利用插件服务库 PlugInsService.dll 封装的功能，干你的需求，你也可以发挥自己的才智，把你处理好的图象，回传到神州易刻。比方说有些地区需要特殊的防伪纹线，但神州易刻设计不出来，那么你只要命令神州易刻导出图片，你获得图片后，就可以自行绘制，绘制好之后再传回神州易刻

}

重要的事再强调： GetPluginsNameAndID 和 PluginsExecute 是你的 dll 插件里你自己至少要编写的两个函数，而不是我方封装的函数。相当于神州易刻提供了一个表格，表格里有两项你必须规范填写，否则你这张表格作废。也相当于 GetPluginsNameAndID 是上学报名，PluginsExecute 就是做作业和考试！

我好象没法动手呀。如果是这样，可能就是因为你忘记了插件服务库，也可能是你还没到可写插件的时候(请果断放弃)！插件服务库 PlugInsService.dll 就是服务你编写插件的。

Dll 插件综论

用什么语言编写插件最好呢？依我个人的观点来看，VC6 是最佳的。可能有人说，VC6 是不是太古老了一些啊，都 20 几年前的了。底层编译器永远是年轻的！一直在变的只是 IDE 和封装库，而底层编译器基本是不太变化的！所以你只要自己会封装，你用 VC6 即可。

不管你用什么语言开发 dll 插件，都要严谨对待，切忌内存泄漏。因为神州易刻加载 dll 插件后，dll 插件就加载到了神州易刻的进程空间，dll 插件不稳定就会导致神州易刻不稳定，dll 插件崩溃神州易刻就会跟着一起崩溃，这会影响神州易刻的信誉，所以开放接口也是有风险的，因为用户可能不知道是插件所导致，而会认为是神州易刻本身的问题。

怎样用程序代码启动神州易刻?

神州易刻因为加密了启动过程，常见的启动进程的代码是启动不了神州易刻的。考虑到二次开发可能需要用程序代码自主启动神州易刻的需求，特提供一种简易方法：用批处理命令启动神州易刻。

用记事本新建一个文件，文件里写入以下内容，然后保存为 LaserDRW.bat。

```
@echo off
start "" "{神州易刻安装路径}\LaserDRW.exe"
exit
```

启动神州易刻的程序代码(静默启动，不会有什么黑东西闪一下):

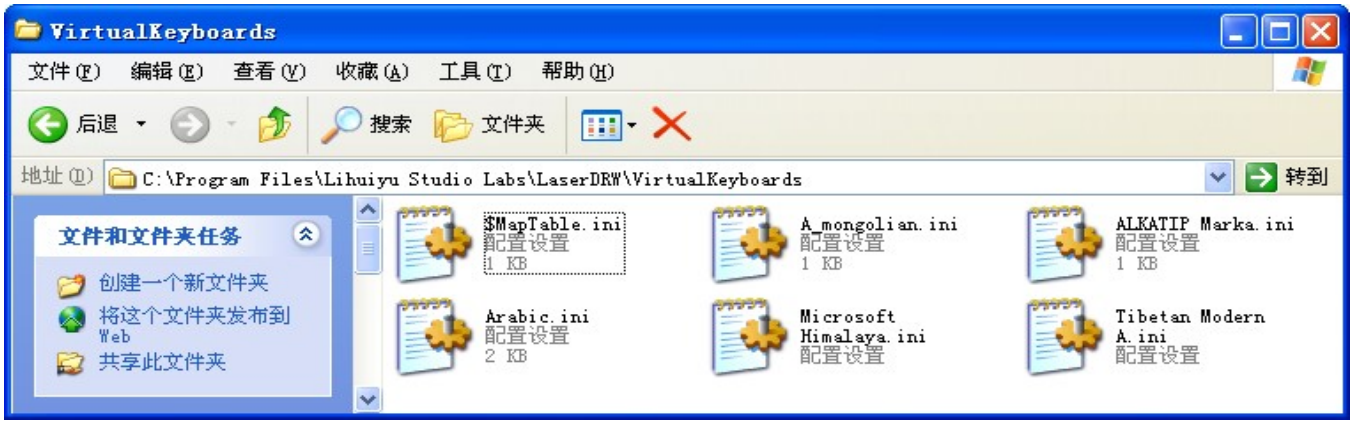
```
ShellExecute(0, "open", "{LaserDRW.bat 的路径}\LaserDRW.bat", NULL, NULL, SW_HIDE);
```

输入软键盘的二次开发

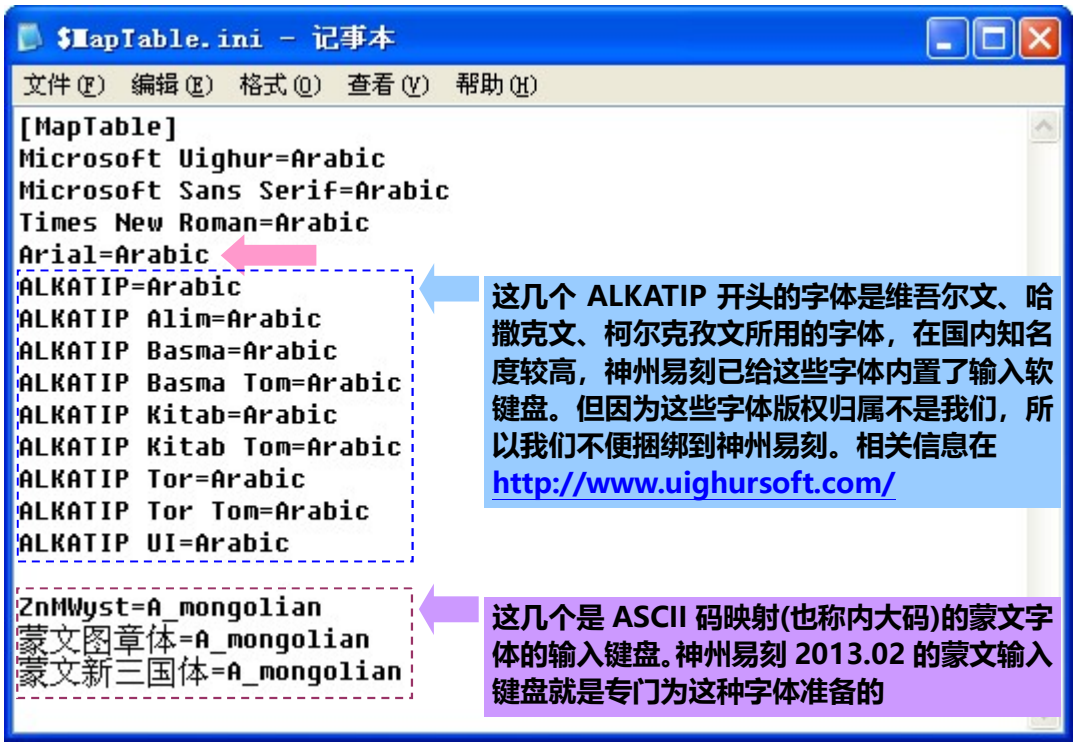
古董版本的神州易刻 2013.02 版本也有一个蒙满文输入软键盘，但其功能十分有限，仅支持输入 ASCII 映射的蒙满文字体，但是，新版本的神州易刻有一个非常强大的文字输入软键盘，而且支持二次开发。如上图所示

神州易刻的输入软键盘：为 Arial 字体建立了维文输入软键盘(当用户选择 Arial 字体会自动创建这个软键盘)

从上图我们可以看到：神州易刻的输入软键盘上共有编号 0 到编号 60 共 61 个按键，每个键又分为上档键和下档键。那么如何制作特定字体的输入软键盘呢？在神州易刻目录下有一个 VirtualKeyboards 文件夹，制作输入软键盘的秘密就在这个文件夹里。



VirtualKeyboards 文件夹里有一些 ini 文件，这些 ini 文件就是输入软键盘的秘密。我们先打开 $MapTable.ini 这个文件(我们称其为映射表)，看看其中的内容。



我们发现$MapTable.ini 有这么一行：Arial=Arabic，这一行指明了 Arial 这个字体使用名为 Arabic.ini 这个文件所定义的键码库！在$MapTable.ini 映射表中，我们发现很多 xxxxx=Arabic，也就是说同一字体族的字体只需要编写一个键码库即可。

注意：字体名与字体文件名不要搞混淆了。字体文件名是可以随意修改的名称，但无论你把字体文件改成什么名字，这个字体文件安装后，其显示的字体名都是固定的！这两者不要混淆了，也就是说输入软键盘使用的是字体名，不是字体文件名！

$MapTable.ini 映射表文件很简单，仅仅是指定某个字体使用哪个键码库。二次开发的重点在键码库的开发。

在 VirtualKeyboards 文件夹里，我们找到键码库 Arabic.ini，打开看看其中的内容。如下图打开了 Arabic.ini

键码库，我们看看其中的内容。

[SETTING] //键盘设定

KB_Count：指定该键盘包含几个键盘

{因为键盘上总共有 61 个按键，如果需要的按键不够用(比如藏文)，那么就可以分页为数个键盘，这时我们就需要设置 KB_Count=n。如果键盘包含 2 个以上的键盘，右下的 Ctrl…键即为切换键盘的键
}

KB_Names：指定键盘的名称，该名称会显示在空格键上

{如果键盘包含几个键盘，那么键盘名字以 | 分隔
比如 KB_Names=维吾尔文(Uighur)|哈萨克文(Kazakh)|柯尔克孜文(Kirghiz)
}

Percent：键盘放大百分比

{神州易刻的输入键盘是矢量化实时绘制的，可以设置键盘显示的大小。Percent 控制键盘放大的百分比
}

CharPercent：键盘上键符放大百分比

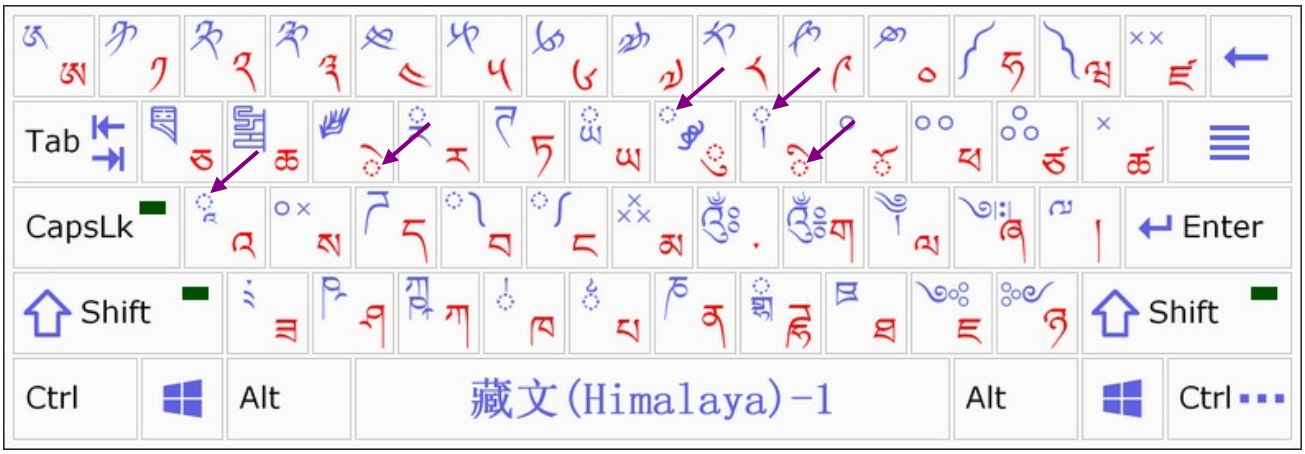
{不同字体的符号绘制到键盘的按键上，有可能太大或太小，可以设置键符绘制的百分比，使按键上绘制的键符美观、协调、清晰
}

Menu_1：竖排键符，取值为 0 或 1，默认值是 0

{如果 Menu_1 =1，那么键符竖排，该特性主要用于蒙(满)文的输入键盘
提示：一共有 Menu_0/ Menu_1/ Menu_2/ Menu_3/ Menu_4/ Menu_5，控制键盘的特性，可点击 Enter

键上方的控制菜单，了解一下，我们在此不赘述。
}

PrefixChar：前导符号(16 进制内码)
{何谓前导字符呢？因为某些特殊字体的一些字符，是放在前一个字符的上面或下面，也就是说它们前面若没有字符，那么这些必须放在前一个字符上面或下面的字符，就不会显示或显示错误。这个前一个字符就是前导字符。前导字符有两个作用，其一是使键符能够正确地显示；其二指明该字符与前面一个字符的关系(在前一个字符上面还是下面)。如下图的藏文键盘上的一个个虚线圆圈(实际是内码为 25CC 的字符)，就是前导字符。前导字符仅用于辅助键符正确显示，点击它的按键，前导字符是不会输出的



如果不设定前导字符，那么系统内定前导字符是：PrefixChar= 25CC，但是，并非所有字体中都有内码为 25CC(虚线圆圈)的字符，若没有 25CC 字符，就要指定一个合适的前导字符。前导字符必须包含在该字体中，不能指定一个该字体中没有的字符作为前导字符。
}

Unicode：Unicode 字体还是 Ansi 字体。取值为 0 或 1。默认值是 1
{ Unicode = 1，表明该字体是 Unicode 字体，否则是 Ansi 字体
}

提示：除了可在[SETTING]段配置外，还可以专门为某个字体配置。如果某个字体(比如 Arial 字体)不想使用[SETTING]的统一配置，那么可以增加[字体名]为其配置专用设置，如下图

[Arial]
……

输入键盘的二次开发相对比较简单，因为 VirtualKeyboards 文件夹里有我们制作的键码库文件参考，开发者只要修改其参数测试其功能即可明白其逻辑。下面我们要说下键符如何设置。

[KEYS.0]： 第一个键盘的下档键
[SHIFT.KEYS.0]：第一个键盘的上档键

[KEYS.1]： 第二个键盘的下档键
[SHIFT.KEYS.1]：第二个键盘的上档键
……
[KEYS.n]： 第 n+1 个键盘的下档键
[SHIFT.KEYS.n]：第 n+1 个键盘的上档键

我们示例一下如何分配键符：

```
[KEYS.0]
0=xxxx
1=yyyy
……
```

这样，我们就给 0 号键分配了键符 xxxx，给 1 号键分配了键符 yyyy。键符即对应字符的 16 进制内码。用户可测试 0=5979，16 进制 5979 实际是汉字她的内码。再测试 0=59795976，我们会发现 0 号键上是奶她，因为奶的内码是 5976，她的内码是 5979，也就是说，神州易刻的输入键盘还可以把汉字作为键符，利用这个特性可以设计一个常用生僻字输入的专用键盘。

键符分配很简单，不赘述。下面我们重点说明一些注意事项

1、 每一个键可分配 1 个或 2 个键符，规则是低位字符在前。0=59795976，0 号键上显示的是奶她而不是她奶
2、 如果定义了前导字符 PrefixChar=nnnn，如果 0=5979nnnn，那么 0 号键上我们看见的是两个字符，但按下 0 号键时，只输出她，因为前导字符是不会输出的
3、 0=0nnn0yyy，可以写成 0= nnn0yyy，但不能是 0=nnnyyy 或 0=0nnnyyy，也就是说仅第一个 0 可以省略
4、 有些字体的键符你可能很难找到其内码，因为它看起来是一个字符，实际可能是两个字符的组合体，而且这个组合体是变形的。就类似这样的逻辑：a 单独打出来是 a，b 单独打出来是 b，但是 ab 排在一起，出来的却是毫无联系的 P。所以你要能用键盘打出 P，就要给对应的键分配 0=b 的内码 a 的内码，因为 P 看起来是一个字符，但它并没有独立的内码，而是 b 的内码 a 的内码的组合
5、 与其他软键盘不同的是：神州易刻的软键盘切断了实际键盘之间的联系，也就是说用户只能用鼠标点击屏幕上的按键输入而不能用实际的键盘输入。这样做的目的是：你仍旧可以用实际的键盘打出 abc123，如果屏幕键盘与实际键盘同步的话，那么需要打 abc123 就不方便了
6、 如何知道字符的内码呢？有两种方案，第一种方法是编写个小程序，把字符强制转换为 16 进制整数，该整数即该字符的内码；另一种方法是在 Word 软件中的插入符号页面把字体设置为指定字体，然后找到你需要的字符，其字符代码即其内码！

新版本的神州易刻内置了 ASCII 映射蒙(满)文的输入键盘、Microsoft Himalaya 藏文输入键盘、维文输入键盘、哈撒克文输入键盘、柯尔克孜文输入键盘、Tibetan Modern A 藏文输入键盘。新版本神州易刻的输入键盘是可以二次开发扩展的，这才是其最有意义的改良：比如要使神州易刻支持某特殊语言文字的输入与排版，不一定需要我们升级神州易刻，因为其他人也可以去做这件事！

结语(附加业务价值：模板库、键码库、ini 脚本配置、外部协作程序、各种 dll 插件、适配其他主板……)

新版本的神州易刻安装包仅仅 2.35M，但这 2.35M 还包含了一些图片、字体、语言包、教学模板库、键码库、驱动程序包等许多东西，也就是说神州易刻程序部分，实际上只有 1.5M 左右，可见其代码之精练高效。而且神州易刻还能完美支持 Windows XP/7/8.0/8.1/10/11(未放弃一个 Windows 系统)，用户既可以在配置落伍的古老的计算机上流畅运行神州易刻，也可以在时髦的 Windows 11 上使用神州易刻。

(完)

2024 年 6 月 05 日

李辉宇

杭州宇骐科技有限公司