Software Tools

Introduction:

While all the tools in this document were developed between 2019 and 2024, they are copywrited from 1998 to 2024 because SoftGlue was first formed in 1998 and some of the tools listed here were first thought of and framed between 1998 and 2008 when the first SoftGlue development was stopped; such as the RGB Dynamic color Palette, the Embedded Syntax Translator & the encrypted dynamic Hexadecimal to Decimal algorithm API.

Definitions:

Software Tools:

Full Application:

An application that is general purpose solution API to any application or a developer's tool application.

1) Embedded Syntax Translator
2) RGB Dynamic Color Palette
3) Visual Powershell Script Compiler
4) Powershell Transposed Visual C Binary Compiler (VCC) - (not developed)

Specific API Application or Script:

A unique full secondary application which by itself serves no purpose or a unique specific script API function:

1) Application Registry API (ARA) reference object
2) Browser Thread Executable (BTEXT)
3) Hexadecimal to ASCII Theorem Algorithm
4) Matching ASCII to Hexadecimal Algorithm

SoftGlue Software Tools:

1) Application Registry API (ARA) reference object:

The ARA is a dynamic encryption application registration and startup API. It with the service of a hardcoded startup encrypted key registers the application through the installation program in the HKEY_CURRENT_USER as encrypted hexadecimal text, after installation is complete with the assistance of the Installation Operating System Command API. The current internal DevOps automated ARA API interface, but will after funding become a licensed non global dynamic link library (DLL), then performs an application root folder verification check. If passes, application starts. If fails returns a startup run time error and the application ends its process and quits. ARA API interface must be in each registered application root folder for fundamental operating system and anti-infringement licensing reasons of the intended registered application.

The ARA fully supports a dynamic 30 day trial timer & a 25 character unique license key obtained from a third party license key supplier. This is one of the limitations of the ARA. It only supports a 25 character license key, for maximum license licensable applications protection against infringement. The 30 day timer is dynamically obtained after initial installation. It's unique to each individual installation because it obtains the exact computer date and time and encrypts it and records it in the applications root folder of the HKEY_CURRENT_USER registry database. The user can at any time within the 30 say trial enter the license key into the application specific key entry calling the ARA API interface, root folder bind DLL.

It's important to mention that all the applications identity information is encrypted except for the identity value, which is the applications name for legal licensing & anti-infringement reasons. However, its text data is encrypted. It is in this way, the SoftGlue license holder of the Application Registry API (ARA) reference object is both legally & technology protected against infringement theft and license fraud. See ARA Register image, ARA Registry Image of a SoftGlue computer application installation registry settings. See ARA Registry Image for details.

2) Browser Thread Executable (BTEXT):

This API is a client to remote internet browser interface loading the clients intended HTML page into the intended browser. The input to the BREXT is completely encrypted. It supports default and specific chosen client browser control. I also supports all currently supported main stream browsers, MS Edge, Mazola Firefox, Google Chrome & CCleaner Bowser, browsers. BREXT is used with all published SoftGlue applications to call their online help document, except the CMDThread. It simulates script control as a command console executable.

3) Embedded Syntax Translator (EST):

The EST transforms any text file of standard text formats as visual basic 6 and visual basic 2022 .net platforms and creates either a temporary file or bas module for importing into Visual Studio. With the EST you can append as many text files as the developer wants, creating a callable script subroutine for each one. The EST solves the developer problem of embedded foreign script into a unique and

SoftGlue Software Tools(Continued):

different development platform allowing that platform to redeploy that embedded foreign script. If the business licensing is approved it will be possible for the developer to embed any script text file(s) into a single callable Visual Basic 202x .net executable and as VB20x .net the developer can then deploy that executable over a secured remote TCP / IP network for remote IT or even web hosting administration. Its primary development purpose is to support script compilers such as Azure and GitHub java script web hosting applications. These scripts can now be called form VB20x .net executable binary, redeploying those scripts to run in local internet browser memory, as they do now as script tagging memory buffering management. Deploying GitHub and other Microsoft 3rd party web hosting scripts are now completely callable by a Microsoft development VB20x .net tool host, greatly improving the internet experience security. However, it can also embed as long and as complex any powershell script for redeployment. There are two examples of this capability to advertise the EST tool as redeployment of embedded script, which is actually encrypted to scripts themselves, but is readable by the developer, called the Visual Powershell Repository Compiler, full script compiler that runs in Powershell Runtime Environment (PRE), and a complete video focused & indexed threaded file system compiler application demo and Web Links an general purpose internet scrobbler that calls an external proprietary client to internet browser API interface called the Browser Thread Executable API to load the clients specific HTML link for reading. The EST also as believed strong second software tools marketing. It can be used to easily DevOps Automate any legal document text as a binary executable, as file.txt or as file.rtf rich text. It embeds the rtf file as its rtf script. The EST can also embed the rtf encrypted images as their rtf text and re-deploy it as well.

An example of this is the Softglue license document. It's a notarized rtf formatted legal document. The license application called Lic.exe loads the embedded legal document into wordpad as an exact copy of the original, then discards it after wordpad is closed. See the online EST help document for its usages; https://softglue.net/wp-content/uploads/2024/02/EBSTHelp.pdf.

4) Hexadecimal to ASCII Theorem Algorithm & Matching ASCII to Hexadecimal Algorithm

This algorithm is entirely written as script logic theorem solution. It is not a binary ALU foating point solution such as the Microsoft Bags / BagMRU related encryption solution. Its pure text hexadecimal encryption & decoding. It's the core solution to the **Application Registry API (ARA) reference object** file system security solution. However, for it to be an OS standard API reference it has to be licensed as double encryption by the OS Company. The OS company needs to provide a Bags / BagMRU second encryption or apply the developers encrypted license the same as it does to all licensed binaries to prevent decompiling to prevent anyone who has the Hexadecimal to ASCII Theorem Algorithm license from decrypting infringement of another's registry application's information. Esently then you can only decrypt your own encryptions, making those encryptions secure as they currently are for the currently posted applications because SoftGlue currently is the only company with such a Hexadecimal to ASCII Theorem solution. See ARA Registry Image as a direct example.

SoftGlue Software Tools(Continued):

5) RGB Dynamic Color Palette Object Reference (RGBDCPalette):

The RGBDCPallete is currently a standalone executable. It's intended to be developed as an embedded-able object or as a dynamic link library (dll) for external graphics development application interface, returning its final color and font formatting as graphics choice of color and font formatting combinations for artist graphics art. Currently this is only possible with a static picker Hume which only returns color & returning fine color variations can be difficult with the mouse control. The graphics artist has to independently transfer between color and texting tools to get their final desired art result. The RGB Dynamic Color Palette, allows the graphics artist to perform each task simultaneously with 100% interactive control & the graphics artist has exact color picking control with Red, Green & blue index control from 0 to 255 for each. It will always be offered as a standalone graphics tool application for the independent artist & software developer, but the graphics developer's applications API solution is a key business unit to the RGBDCPalette graphics software applications business. See online https://softglue.net/wp-content/uploads/2024/02/RGBDynamicPaletteHelp.pdf help for details of its usages.

6) Operating System Common Dialog Executable (OSCMDALG):

This API reference object executable is only an application of the Visual Studio 2022 common dialog open and save application development references that are each bind to a transparent form as development references. It uses traditional windows 3.11 ini formatting file transfer to communicate with the application calling it to access the file system for user input. Its purpose is to reduce the file system kernel noise the common dialog development reference naturally creates increasing the overall final application OS kernel(s) noise level due to the facts that the common dialog reference has to pause the application, as an external application while the user is accessing the file system to obtain the file path needed to continue. Using OSCMDALG to access the file system and returning the same result, allows the intended application to pause itself with standard running processes control logic, greatly reducing the common dialog runtime noise. Thereby, reducing the natural total application kernel noise generated by the intended application. The kernels are lead to believe that the user / application just started another independent application in which its only memory management by the kernels, which is entirely normal kernel operations has to manage not affecting the overall operating system runtime noise, which cause disruptions in its own OS management of services and updates. The OSCMDALG also supports vb script, BATCH & Command scripts for local server administration. The script can easily call the OSCMDALG object reference with the traditional exec run command and use its status bit as switch to tell it when the OSCMDALG process has ended. This is an traditional windows 3.11 process control API function, outdated with 64-bit technology development, but is still allowed with traditional 16 & 32-bit scripts, used for certain server administrations. A very much needed technical support as of current understanding by this documents author. It's currently done with tradition means in various ways, but sometimes it's not possible because of the limitations of the common dialog reference object as traditional direct use. It is believed by softglue that the OSCMDALG greatly solves this user interactive server problem giving much more server administration user interactive flexibility. The online CMDThread demo performs a simple application to common dialog look up with the OSCMDALG executable and returns it as an ini

SoftGlue Software Tools(Continued):

file returned result then read by the intended application, CMDThread demo; displaying the result as a final application message. See the online OSCMDALG help document for instructions on how to perform this; https://softglue.net/wp-content/uploads/2024/02/OpenSaveDialogHelp.pdf.

7) Visual Powershell Script Compiler (VPWSC):

The VPWSC is full powershell language script compiler of the MSDN forms objects, events, methods and properties. It adheres to industry stands of software definitions and notation and its rigorous defined architecture with compiler development rules, as all compilers are. It runs in a proprietary powershell manifest &powershell script module RAM memory block Powershell Object Notation or PSON thread. The PSON is analogous to the command line JavaScript Object Notation or JSON. It performs the same memory block functions of reserving callable memory space for each script function and its variables, allowing any script function with the same memory block to call it, thereby building a complete application memory space the same as an executable. There are two SoftGlue.net offered examples of the working Visual Powershell Script Compiler; the Visual Powershell Repository Compiler Demo (VisualPWSDemo) and Web Links.

1) VisualPWSDemo is an example of a full video focused & full index threaded file system application. I can call an unlimited of ended forms and keep track of each form's functions, properties and values it returns, exactly the same as any currently licensed compiler such as C++ or Visual Basic. Among other proprietary things the VPWSC has its own completely indexed Toolbar Container and Toolbar strip API algorithm that supports as API function calls all MSDN toolbar objects & are then modifiable in the same manner as the toolbar objects are modifiable with current VS2022 C++ or Visual Basic once those objects are created by the new VPWSC Toolbar object API reference. Its architecture is that of a working column and row script spread sheet allowing any API call to keep track of the runtime toolbar's header list and each header's list contents as columns and row calls. See https://softglue.net/wp-content/uploads/2024/02/VPWSDemoHelp.pdf for complete description details.

2) Web Links is a client applet example of a fully functional partial video focus control that calls an external visual basic 2022 .net browser interface application to deploy any called HTML page in the browser. It could have easily written with any other standard language such as Visual Basic, C++ or even C sharp. Its serves the purpose of demonstrating that the Visual Powershell Script Compiler then once transposed Visual C compiler is practical viable developers optional language as part of the overall developers development tool kit. See https://softglue.net/wp-content/uploads/2024/02/Web-LinksHelp.pdf for complete description details.

SoftGlue Software Tools(Continued):

8) Powershell Transposed Visual C Binary Compiler (VCC) - (not developed):

The purpose of developing the VPWSC was to develop the foundation for a transposed powershell console Visual C compiler, using the PSON as its fundamental thread communication between, C classes, functions, subroutines, events, methods, properties & constants. This is something since the development of windows 95 of the early 1990's has been highly desired by all software development including the operating system companies such Microsoft & it's been attempted by them on many occasions, with no runtime security foundation success. See Visual C Powershell Binary Compiler for full discussion. It's for this main reason C++ a full API thread-able C language & compiler was developed. But it has fundamental command line console security limitations and resulting technology politics sounding it, with both the final developed software product's usage and the development process itself with Visual Studio 202x. A successful Powershell console Visual C Compiler with .net support will not have these technology security issues & its technology issues will be no different than its augmented counterpart Visual Basic 2022 .net. See the Visual Powershell Repository Compiler help for complete description details; https://softglue.net/wp-content/uploads/2024/02/VPWSDemoHelp.pdf

ARA Registry Image: