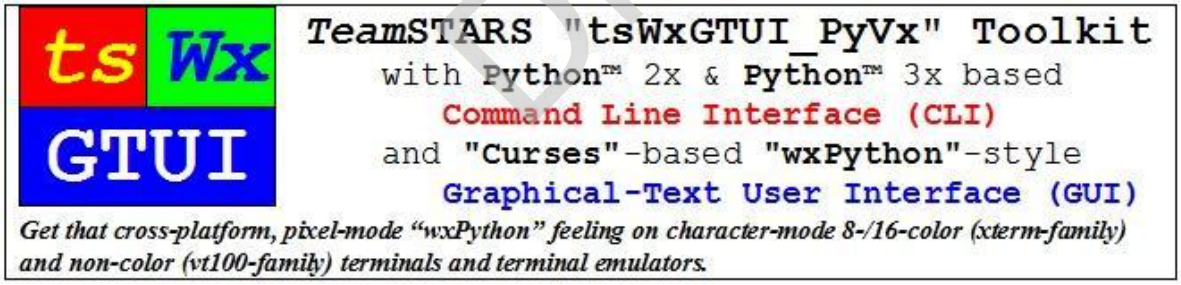
Software Gadgetry

Announcement

Vol. 0 - "tsWxGTUI" Toolkit

Rev. 0.0.1 (Pre-Alpha)

Author(s): Richard S. Gordon



Printed on 6 August, 2015

Contents

1 About

The *Team*STARS "tsWxGTUI_PyVx" Toolkit is a free and open source cross-platform Python framework for writing user friendly Command Line Interface (CLI) and Graphical User Interface (GUI) applications.

1.1 Cross-Platform Application Programming Interface (API)

The *Team*STARS "tsWxGTUI_PyVx" Toolkit's cross-platform Application Programming Interface (API) insulates your application design from differences in host operating systems and operator terminals.

Your applications will:

1. Work without change on the most popular and readily available 32-/64-bit computer server and desktop platforms.
   a) Your character-mode CLI applications can be implemented in the Python 2x and/or Python 3x generation programming languages. CLI applications can:

      (1) use a standard computer terminal display to prompt the operator for input and output any associated response;

      (2) use a standard computer terminal keyboard to receive operator commands and data.

   b) Your character-mode GUI applications can be implemented with a subset of the popular pixel-mode cross-platform wxPython API. GUI applications can:

      (1) use a standad computer terminal display to create and update tiled and overlapping context-dependent frames and pop-up dialogs to prompt the operator for input and output any associated response;

      (2) use standard computer terminal keyboard and mouse (trackball, touch pad or touch screen) to receive operator input.
2. Operate in an isolated local system or in conjunction with a remote networked system.

2 Platforms

Application programs can use both the CLI and GUI portions of the TeamSTARS "tsWxGTUI_PyVx" Toolkit's framework ONLY WHEN the platform's Python 2x and/or Python 3x have access to either the Curses or nCurses Terminal Interface library. Without such access, application programs can only use the CLI portion of the framework.

(1) GNU/Linux platforms (Debian-based and others with the GNU toolchain and Linux kernel) and native releases of Python 2x and/or Python 3x typically include access to the nCurses Terminal Interface library. The mouse is fully supported (by single, double and triple clicking) for 8-/16-color xterm-family terminals and partially supported (by single clicking) for non-color vt100-family terminals.

(2) Mac OS X (Darwin and BSD Unix-based) platforms and native releases of Python 2x and/or Python 3x typically include access to the Curses Terminal Interface library. The mouse is fully supported for 8-/16-color xterm-family terminals but unsupported (ignores all clicking) for non-color vt100-family terminals.

(3) Windows (NT-based) platforms and native releases of Python 2x and/or Python 3x NEVER include the necessaryTerminal Interface library access. To work around this Microsoft Windows platform limitation, users should download, install and use Cygwin, the free GNU Toolchain and Linux-like plug-in from Red Hat, and its alternate Python 2x and 3x support. The mouse is fully supported (by single, double and triple clicking) for 8-/16-color xterm-family terminals and partially supported (by single clicking) for non-color vt100-family terminals.

(4) Unix (AT&T, BSD, Darwin, FreeBSD, Solaris and other Unix-based) platforms and native releases of Python 2x and/or Python 3x typically include access to the Curses Terminal Interface library. The mouse is fully supported for 8-/16-color xterm-family terminals but unsupported (ignores all clicking) for non-color vt100-family terminals.

The TeamSTARS "tsWxGTUI_PyVx" Toolkit has been designed and is known to work on 32-/64-bit computers with keyboard and mouse input and color and/or monochrome display console terminals on the following host platforms.

2.1 GNU/Linux

Developed and/or tested using the "Terminal" application with non-color (vt100, vt220), 8-color/64-color pair (xterm, xterm-color), 16-color/256-color pair (xterm-16color, xterm-88color and xterm-256color) terminals/terminal emulators on the following GNU toolchain and Linux kernel-based operating system desktop distributions:

1. CentOS 7 Linux last used in 2015

2. Debian 8 Linux last used in 2015
3. Fedora 22 Linux last used in 2015
4. OpenSuSE 13.2 Linux last used in 2015
5. Scientific 7 Linux last used in 2015
6. Ubuntu 15.04 Linux last used in 2015

2.2 Mac OS X

Developed and/or tested using the "Terminal" or third-party "iTerm" and/or "iTerm2" application with non-color (vt100, vt220), 8-color/64-color pair (xterm, xterm-color), 16-color/256-color pair (xterm-16color, xterm-88color and xterm-256color) terminals/terminal emulators on the following Darwin and BSD Unix kernel based operating system desktop distributions:

1. Mac OS X 10.4 "Tiger" last used in 2007 with "iTerm"
2. Mac OS X 10.5 "Leopard" last used in 2009 with "iTerm"
3. Mac OS X 10.6 "Snow Leopard" last used in 2011 with "iTerm"
4. Mac OS X 10.7 "Lion" last used in 2014 with "iTerm2"
5. Mac OS X 10.9 "Mavericks" last used in 2014 with "iTerm2"
6. Mac OS X 10.10 "Yosemite" last used in 2015 with "iTerm2"

2.3 Microsoft Windows (with Cygwin plug-in)

Developed and/or tested using the Cygwin, the free GNU/Linux-like plugin from Red Hat, and its "mintty" application with non-color (vt100, vt220), 8-color/64-color pair (xterm, xterm-color), 16-color/256-color pair (xterm-16color, xterm-88color and xterm-256color) terminals/terminal emulators on the following GNU Toolchain and NT-kernel based operating system desktop distributions:

1. Windows XP Professional last used in 2014

   Use of this platform for development and/or testing ended on 8 April 2014 when Microsoft ceased issuing security updates, bug fixes and product enhancements.
2. Windows 7 Professional, last used in 2015
3. Windows 8 Professional, last used in 2014
4. Windows 8.1 Professional, last used in 2015

5 Windows 10 Professional, last used in 2015

2.4 Unix

Developed and/or tested using the "Terminal" application with non-color (vt100, vt220), 8-color/64-color pair (xterm, xterm-color), 16-color/256-color pair (xterm-16color, xterm-88color and xterm-256color) terminals/terminal emulators on the following Unix-based operating system desktop distributions:

1 OpenSolaris 11 Unix, last used in 2013

2 OpenIndiana 151a8 Unix (Open Solaris 11-based), last used in 2014

3 PC-BSD 11 Unix (FreeBSD-based), last used in 2015

3 Framework

The *Team*STARS "tsWxGTUI_PyVx" Toolkit's framework includes:

1. tsLibCLI --- Libraries of Python CLI building-block modules
2. tsLibGUI --- Libraries of Python GUI building-block modules
   a) use the most popular 8-/16-color (xterm-family) and non-color (vt100-family) standad computer terminal displays to:

      (1) create and update context-dependent frames with optional menu bars, status bars, text panels, buttons, scrollbar positioning arrows

      (2) create and update context-dependent pop-up dialogs with optional checkboxes, radio buttons;

   b) use standard computer terminal keyboard and mouse (trackball, touch pad or touch screen) to receive operator input.
3. tsToolCLI --- Python CLI tools and application programs
4. tsUtils --- Shell script tools
   a) library includes both Python 2x and Python 3x programming language generation frameworks.
   b) In the Python 2x and/or Python 3x generation programming languages.
   c) Its GUI design is implemented with the Python Curses console terminal interface library.
   d) Its GUI design is implemented with the Python 2x and/or Python 3x Curses terminal interface packages.

      It emulates a character-mode compatible subset of the popular cross-platform pixel-mode "wxPython" API.

      It enables you to write applications for the most popular 8-/16-color (xterm-family) and non-color (vt100-family) console terminals.

   e) It can operate in an isolated system (Stand-Alone mode) or in a networked system (Stand-Among mode).
5. It comes with an extensive collection of easy to read and modify source code and examples.
   a) Its CLI and GUI subsystems and component design facilitates development, maintenance, troubleshooting and porting the implementation for use with newer or older platforms which are less popular or not as readily available.
   b) Its developer-sandbox design facilitates development, maintenance and troubleshooting of both the toolkit foundation and your applications.
   c) Its site-package design facilitates deployment of both the toolkit foundation and your applications.
6. It comes with an extensive collection of documentation, manpages and notebooks.
7. Is free for use, modify and redistribute in both open source and commercial applications.

3.1 tsLibCLI

The *Team*STARS "tsWxGTUI_PyVx" Toolkit software provides the following:

1 **tsApplication** - Python base class to initialize and configure the application program launched by an operator. It enables an application launched via a Command Line Interface (CLI) to initialize, configure and use the same character-mode terminal with a Graphical-style User Interface (GUI).

a) Input provided on the command line by an operator. The command line uses a Unix-style, free-format to promote future enhancement and on-going maintenance.

b) Input provided in the parameter list of the application's invocation of the class instantiation. The parameter list uses a Python-style free-format to promote future enhancement and on-going maintenance.

2 **tsCommandLineEnv** - Python class to initialize and configure the application program launched by an operator. It delivers those keyword-value pair options and positional arguments specified by the application, in its invocation parameter list. It wraps the Command Line Interface application with exception handlers to control exit codes and messages that may be used to co-ordinate other application programs..

3 **tsCommandLineInterface** - Python class establishes methods that prompt or re-prompt the operator for input, validate that the operator has supplied the expected number of inputs and that each is of the expected type.

4 **tsCxGlobalsPkg** - Python class to provide a theme-based centralized mechanism for modifying/restoring those configuration constants that can be interogated at runtime by those software components having a "need-to-know". The intent being to avoid subsequent searches to locate and modify or restore a constant appropriate to the current configuration, users and their activities.

5 **tsDoubleLinkedList** --- Class to establish a representation of a linked list with forward and backward pointers.

6 **tsException** - Python class to define and handle error exceptions. It maps various error exceptions into an 8-bit exit code and message that can be used to coordinate the actions of a set of shell scripts.

7 **tsLogger** - Python class to define and handle event message timestamping, formatting and output. It also supports logging of assert and check case results.

It supports the following message severity levels: NOTSET (lowest priority), DEBUG, INFO, NOTICE, WARNING, ALERT, ERROR, CRITICAL, EMERGENCY (highest priority) and PRIVATE.

It defines and handles event message processing associated with the following: wxPython/wxWidget exceptions: wxASSERT, wxASSERT_MSG, wxCHECK, wxCHECK2, wxCHECK2_MSG, wxCHECK_MSG, wxCHECK_RET, wxFAIL, wxFAIL_COND_MSG, wxFAIL_MSG and wxTRAP.

8 **tsOperatorSettingsParser** - Class to parse command line options and arguments.

Supports one or more of the parser module(s) available in the Python version(s) supported by the application.

a) "argparse" (introduced with Python 2.7.0)

b) "optparse" (introduced with Python 2.3.0)

c) "getopt" (introduced with Python 1.6.0)

9 **tsPlatformRunTimeEnvironment** - Class to capture current hardware and software information about the run time environment for the user process.

10 **tsReportUtility**- Class defining methods used to format information such as time, date, data size, elapsed time and nested dictionary contents.

11 **tsSysCommands** - Class definition and methods for issuing shell commands to the host operating system.

3.2 tsLibGUI

NOTES:

1) As a character-mode GUI-style toolkit, this product does not support those "wxPython" features associated with graphical elements such as bit images, icons, proportional-spaced fonts or HTML and XML text markup. The current release supports the porting of "wxPython" 2.8.9.2 application(s) to platforms running Python 2.5.x, 2.6.x, 2.7.x, 3.1.x, 3.2x, 3.3x and 3.4.x.

2) Technical and resource issues drive the development effort. Initially, development focussed on establishing the feasibility of emulating core components of the the "wxPython" and associated "wxWidgets" API. Development then iteratively seeks to establish usability-enhancing components and to then identify and resolve any appearance, behavior and API-conformance issues.

3) See the unpublished **"tsWxGTUI_PyVx" Vol. 7 - Design Notes** (it was created only for personal use) for the identification and an overview of those currently implemented class modules, ones currently under construction and ones for which development applicability and/or commitments are still TBD.

The *Team*STARS "tsWxGTUI_PyVx" Toolkit's High-Level Widget API identifies the basic wxPython-style Graphical User Interface features that the Software Engineer can include for input and output interactions with the System Operator. It includes such widgets as the following:

1 **Button** - GUI object that may contain text or character-mode icons. If a mouse button is clicked when the cursor is over the object, an associated function or method will be initiated. The function or method will send a signal to notify other GUI objects of the event. Buttons are clickable windows that trigger an associated event (such as start, pause, resume or terminate an operation).

2 **Carat** - GUI object that may contain a set of special character-mode symbols, usually including a solid rectangle or a blinking underline character, showing the position where the typed text will appear.

3 **Check Box** - GUI object that may contain text or character-mode icons. If a mouse button is clicked when the cursor is over the object, it initiates an associated function or method. The function or method will toggle the check box to the next "ON", "OFF" or "TRI-STATE" value. The function or method will send a signal to notify other GUI objects of the event. Checkboxes are buttons to toggle the enabled or disabled state of an associated feature (such as start or stop logging)

4 **Cursor** - A special character-mode symbol, usually a solid rectangle or a blinking underline character, that signifies where the next character will be displayed on the screen.

5 **Dialog** - A GUI object with a title bar and sometimes a system menu, which can be moved around the screen. It can contain controls and other GUI objects and is often used to allow the user to make some choice or to answer a question. Dialogs are pop-up windows associated with an application task's menu bar (such as an input field for an operator's search criteria and an output field for a list of candidate WEB sites) that will be terminated upon completion.

6 **Frame** - GUI object whose size and position can (usually) be changed by the user. It usually has thick borders and a title bar, and can optionally contain a menu bar, toolbar and status bar. A frame can contain any GUI object that is not a frame or dialog. Frames are windows associated with an application task (such as an internet WEB Browser) that can be minimized into an icon, expanded to full screen or terminated upon operator demand.

7 **Gauge** - GUI object whose horizontal or vertical bar shows a quantity (often time). It supports two working modes: determinate and indeterminate progress. First is the usual working mode while the second can be used when the program is doing some processing but you don't know how much progress is being done. In this case, you can periodically call the Pulse function to make the progress bar switch to indeterminate mode (graphically it's usually a set of blocks which move or bounce in the bar control). Gauges are used to indicate progress of an associated operation (scale with range such as 0% to %100%).

8 **Menu** - GUI object that features a popup (or pull down) list of items, one of which may be selected before the menu goes away (clicking elsewhere dismisses the menu). It may be used to construct either menu bars or popup menus.

9 **Menu Bar** - GUI object that features a series of menus accessible from the top of a frame. Each operator selectable entry triggers an associated event (such as create a file).

10 **Panel** - GUI object that features a window on which controls are placed. Panels are usually placed within a frame. Its main feature over its parent window class is code for handling child windows and TAB traversal. Panels are container windows for other Lower Level GUI Objects.

11 **Radio Box** - GUI objects that are used to select one of number of mutually exclusive choices. A radio box is displayed as a vertical column or horizontal row of labelled radio buttons. Radio Boxes are a collection of Buttons associated with the same group (such a AM or FM band) that are interdependent such that any one can become activated after the others simultaneously become deactivated.

12 **Radio Button** - GUI object that may contain text or character-mode icons. If a mouse button is clicked when the cursor is over the object, it initiates an associated function or method. The function or method will turn the associated radio button "ON" and turn "OFF" all other radio buttons within the associated radio box. The function or method will send a signal to notify other objects of the event. Radio Buttons are used to activate one of the associated features (such as broadcast channel selection) after the other features associated with the same Radio Box group (such a AM or FM band) have become deactivated.

13 **ScrollBar** - GUI object that includes a horizontal or vertical ScrollBarGauge between two ScrollBarButtons. The operator uses it to re-position (pan) the contents of an associated ScrolledText window.

14 **ScrollBarButton** - GUI object that includes one arrow symbol ("<", ">", "^" or "V") that indicates the horizontal or vertical direction of scrolling. When the operator clicks on a ScrollBarButton, the contents of the associated ScrolledText window moves.

- A single Left Click on one of the two horizontal ScrollBarButtons moves the text by one column. A rapid double Left Click on one of the two horizontal ScrolBarButtons moves the ScrolledText by one page (the horizontal column width). A single Right Click on one of the two horizontal ScrolBarButtons moves the text to the appropriate horizontally end point (either left/right most column).
- A single Left Click on one of the two vertical ScrollBarButtons moves the text by one row. A rapid double Left Click on one of the two vertical ScrolBarButtons moves the ScrolledText by one page (the vertical row height). A single Right Click on one of the two vertical ScrolBarButtons moves the text to the appropriate vertical end point (top/bottom most row).

15 **ScrollBarGauge** - GUI object located between two ScrollBarButtons, contains a bar graph that indicates the position and size of the displayed text relative to the non-displayed text.

- When the bar graph is empty (blank), there is no text available to be displayed.
- When it is completely filled, all of the available text is displayed.
- When it is partially filled, the starting point of the graph displays the starting point of the displayed text relative to the available text. The size of the filled graph displays the size of the displayed text relative to the available text.
- When the operator single Left Clicks on a ScrollBarGauge between its two associated ScrollBarButtons, the contents of the ScrolledText window moves in proportion to the difference between the click and the associated text end positions.

16 **Scrolled** - An application-independent GUI object base class for ScrolledWindow. It lays out the position and size of one ScrolledText window and one or two ScrollBars each with their associated ScrollBarGauge and pair of ScrollBarButtons. It enables an operator to select the columns and or rows of text to be fit and displayed within the available peep hole viewing area.

17 **ScrolledText** - GUI object that allows one or more lines of text to be appended to a retained list. In conjuncton with one or two associated pairs of ScrollBarButtons, it enables an operator to select the columns and or rows of text to be fit and displayed within the available peep hole viewing area.

18 **ScrolledWindow** - GUI object that instantiates an application-specific instance of Scrolled to lay out the position and size of one ScrolledText window and one or two ScrollBars each with their associated ScrollBarGauge and pair of ScrollBarButtons. It enables an operator to select the columns and or rows of text to be fit and displayed within the available peep hole viewing area.

19 **Splash Screen** - GUI object that simulates a pixel-mode bitmap image which is displayed upon application startup. It may contain text or character-mode icons. It features a window with a thin border and text describing the application. The bitmap-like display may be created from Panel, BoxSizer, GridSixer and TextCtrl widgets. It is created and shown during application initialization, before the application's own window. It either explicitly destroyes itself or disappears after a it time-out.

20 **Status Bar** - GUI object that feature a narrow window that can be placed along the bottom of a frame to give small amounts of status information. The object can contain one or more fields, one or more of which can be variable length according to the size of the window.

21 **Static Text** - GUI object that features a text control that displays one or more lines of read-only text.

22 **Text Ctl** - GUI object that features a text control which allows text to be displayed and edited. It may be single line or multi-line. The text may be indented, line-wrapped and scrolled to fit the available display area.

23 Tool Bar (Future) - GUI object that features a series of tool entries accessible from the top of a frame. Each operator selectable entry triggers an associated event that starts the selected tool in a new Frame.

NOTES:

1) Event Handling support currently associates mouse clicks with the enclosing GUI object that is not obsucred by overlaying GUI objects. The toolkit generates an event notification and dispatches it to the event handler designated by the application. It will dispatch unhandled event notifications to a default handler.

2) Keyboard Input Events are detected. The raw characters are echoed but are NOT currently forwarded to the window object having focus.

3) Mouse Input Events are detected. The x-y coordinates and button state are echoed. Single Left Clicks, Double Left Click and Right Clicks are forwarded to the window object positioned to receive and process the event. More comples GUI event detection, analysis and processing is not yet supported. Platform-specific vt100/vt220 terminal emulators do NOT conform to the xterm/xterm-color mouse click event protocol recognized by nCurses. Instead of a single mouse click event, the vt100/vt220 terminal emulators generate a string of escape sequence events. It is unknown if the "tsWxGTUI_PyVx" Toolkit can develop logic to synthesize an xterm/xterm-color mouse click event from the string of vt100/vt220 escape sequence events.

4) Automatic layout support is currenlt provided by the BoxSizer and GridSizer widgets or by combinations of them. More comples GUI object positioning and sizing is not yet supported.

3.3 tsToolsCLI

The *Team*STARS "tsWxGTUI_PyVx" Toolkit software provides the following:

1 tsLinesOfCodeProjectMetrics - Python application program, with a Command Line Interface (CLI), that generates reports of software project progress and the estimated cost (or contributed value) of the project when it is finally completed.

a) It scans an operator designated file directory tree containing the source files, in one or more programming language specific formats (such as Ada, Assembler, C/C++, Cobol, Fortran, PL/M, Python, Text, and various command line shells).

For each file, it accumulates and reports the total number of code lines, blank/comment lines, words and characters.

For each programming language format, it accumulates and reports a summary of details of the associated source files.

For the entire set of source files, it accumulates and reports a summary of details.

It uses the summary of the entire set of source files to derive, analyze, estimate and report metrics for the software development project (such as labor, cost, schedule and lines of code per day productivity).

b) It supports one or more of the parser module(s) available in the Python version(s) supported by the application.

"argparse" (introduced with Python 2.7.0)

"optparse" (introduced with Python 2.3.0)

"getopt" (introduced with Python 1.6.0)

When used with Python 2.7 or Python 3.2, the "tsOperatorSettings.py" module (a "tsLibCLI" component of the "tsWxGTUI_PyVx" Toolkit) supports the current and legacy Python version specific parser module(s) as an experimental and educational opportunity.

However, when one seeks to back port applications to Python 2.0-2.6 or Python 3.0-3.1, the "tsOperatorSettings.py" module must be stripped of unsupported Python version specific parser modules in order to prevent a program trap which will block the application from running.

---

2 **tsPlatformQuery** - Python application program, with a Command Line Interface (CLI), that captures current hardware and software information about the run time environment for the user process.

(a) Host processor hardware support includes various releases of x86, PowerPC, SPARC.

(b) Host operating system software support includes various releases of Cygwin, Linux ('Debian', 'Fedora', 'mandriva', 'redhat', 'Scientific / Centos', 'SuSE'), Mac OS X, Microsoft Windows ('XP', 'Vista', '7', '8', '8.1') and Unix ('FreeBSD / PC-BSD', 'IRIX', 'OpenIndiana', 'Solaris / SunOS').

(c) Host virtual machine software support includes various releases of Java and Python.

---

3 **tsStripComments** - Python application program, with a Command Line Interface (CLI), that transforms an annotated, development version of a directory of sub-directories and Python source files into an unannotated copy.

The copy is intended to conserve storage space when installed in an embedded system. The transformation involves stripping comments and doc strings by de-tokenizing a tokenized version of each Python source file. Non-Python files are trimmed of trailing whitespace.

---

4 **tsStripLineNumbers** - Python application program, with a Command Line Interface (CLI), that strips line numbers from source code (such as annotated FORTRAN program listings) that (unlike BASIC program listings) do not reference line numbers for conditional branching.

Output from fixed format FORTRAN (F77) code is NOT corrected to ensure that each statement first character begins in column 7 and that each ampersand ("&") continuation character is in column 6.

---

5 **tsTreeCopy** - Python application program, with a Command Line Interface (CLI), that copies the contents of a source directory to a target directory.

6 **tsTreeTrimLines** - Python application program, with a Command Line Interface (CLI), that copies the contents of a source directory to a target directory after stripping superfuous white space (blanks) from end of each line.

4 Copyright & License

Author Copyrights & User Licenses for "tsWxGTUI_Py2x" & "tsWxGTUI_Py3x" Software & Documentation

- Copyright (c) 2007-2009 Frederick A. Kier & Richard S. Gordon, a.k.a. *Team*STARS. All rights reserved.
- Copyright (c) 2010-2015 Richard S. Gordon, a.k.a. Software Gadgetry. All rights reserved.
- GNU General Public License (GPL), Version 3, 29 June 2007
- GNU Free Documentation License (GFDL) 1.3, 3 November 2008

Third-Party Component Author Copyrights & User Licenses

- Attribution for third-party work directly or indirectly associated wth the *Team*STARS "tsWxGTUI_PyVx" Toollkit are detailed in the "COPYRIGHT.txt", "LICENSE.txt" and "CREDITS.txt" files located in the directory named "*./tsWxGTUI_PyVx_Repository/Documents"*.

5 Further Information

You can learn more about the*Team*STARS "tsWxGTUI_PyVx" Toolkit by browsing its collection of documentation, manpages, notebooks, source code and examples at:

https://github.com/rigordo959/tsWxGTUI_PyVx_Repository