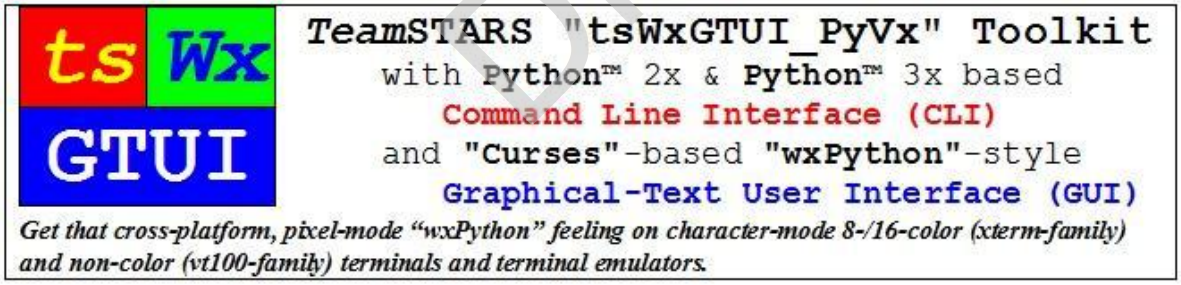
Software Gadgetry

Announcement

Vol. 0 - "tsWxGTUI_PyVx" Toolkit

Rev. 0.0.3 (Pre-Alpha)

Author(s): Richard S. Gordon



Printed on 30 September, 2015

Contents

1 ANNOUNCEMENT

The *Team*STARS "tsWxGTUI_PyVx" Toolkit, version 0.0.3 (pre-alpha), has been released, via "GitHub", for you to freely use, study, modify and redistribute.

1.1 About

The *Team*STARS "tsWxGTUI_PyVx" Toolkit's cross-platform virtual machine design and implementation supports a broad assortment of open and proprietary hardware and software platforms.

The Toolkit's Python source code has been developed and tested only with Intel x86 and x64 processors and representative GNU/Linux, Mac OS X, Microsoft Windows and Unix operating system releases.

Its source code has been compiled, interpreted and executed by Python 2x and 3x Virtual Machines developed by the Python Software Foundation (PSF).

The PSF also distributes equivalent Python 2x and 3x Virtual Machines (and the source code to build them) for other processor types and operating systems, some of which are listed below.

All of the Toolkit's human readable source code:

1. Is that part of computer software which most users don't ever see; it's the part computer programmers manipulate to change how a computer "program" or "application" works.
2. Is included in the release for you to freely use, study, modify and redistribute.
3. May be run on any or all of the platforms which satisfy the following Toolkit hardware and software requirements.

1.1.1 Platform Hardware and Software Requirements

1 Computer Central Processing Unit

A single or multi-core 32-/64-bit processor.

2 Basic Computer Terminal Display and Keyboard for Software Engineering Workstation

A pixel-mode multi-font graphics display that supports at least 80 col x 40 row (640 x 480 pixels) and 16,777,216 colors and industry standard terminal emulators.

a) xterm (8-color, 64-color pairs)

b) xterm-16color (16-color, 256-color-pairs)

c) vt100 (1-color ON/OFF, 2-color-pair NORMAl/REVERSE)

d) vt220 (1-color ON/OFF, 2-color-pair NORMAl/REVERSE)

3 Basic Computer Terminal Display and Keyboard for Embedded System Operator

A character-mode single font text display that supports at least 80 col x 40 row (640 x 480 pixels) and industry standard terminal emulators:

a) xterm (8-color, 64-color pairs)

b) xterm-16color (16-color, 256-color-pairs)

c) vt100 (1-color ON/OFF, 2-color-pair NORMAl/REVERSE)

d) vt220 (1-color ON/OFF, 2-color-pair NORMAl/REVERSE)

4 Mouse or Trackball for Software Engineering Workstation and Embedded System Operator

A cross-platform, industry standard mouse or trackball with:

a) two button (left & right)

b) an optional scroll wheel, which can also act as a third (middle) button

c) an optional touchpad or touchscreen with software that can recognize one and two finger gestures such as tap, drag and scroll

5 Computer Operating System

A multi-user (for local and remote access), multi-process (for interacting with multiple applications) and multi-threaded (for sharing platform resources) operating system such as:

a) GNU/Linux on Intel (x86 and x64) and other architectures

Its POSIX-compatible CLI is provided by the GNOME Terminal, KDE Konsole, XTerm and UXTerm applications and a shell such as BASH.

Its Curses-based GUI is provided by the ncurses terminal control library for Unix-like systems.

b) Mac OS X (Darwin Unix-based operating system) on Apple–IBM–Motorola alliance (PowerPC) and Intel (x86 and x64) architectures

Its POSIX-compatible CLI is provided by the GNOME Terminal and iTerm applications and a shell such as BASH.

Its Curses-based GUI is provided by the ncurses terminal control library for Unix-like systems.

c) Microsoft Windows on Intel (x86 and x64) architectures with "Cygwin", the free GNU/Linux-like plug-in from Red Hat.

(NOTE: Its DOS-like CLI is provided by the Microsoft Command Prompt terminal application which CANNOT support a Curses-based GUI.)

Its POSIX-compatible CLI is provided by the Cygwin Mintty (Terminal) application and a shell such as BASH.

Its Curses-based GUI is provided by the ncurses terminal control library for Unix-like systems.

d) Unix on Apple–IBM–Motorola alliance (PowerPC), HP-UX (PA-RISC), IBM-AIX (RS/6000), Intel (x86 and x64), IRIX (SGI/MIPS), Solaris (Sun/SPARC) and other architectures

Its POSIX-compatible CLI is provided by the GNOME Terminal and KDE Konsole applications and a shell such as BASH.

Its Curses-based GUI is provided by the curses or ncurses terminal control library for Unix-like systems.

6 Terminal Control Library

A cross-platform, industry-standard library of functions that manage an application's character-mode text display on character-cell terminals:

a) curses

The traditional library available on Unix-like systems. It provides a platform-independent Application Programming Interface (API) that enables the same application source code to work with proprietary hardware. Features (such as vt100/vt220 mouse and xterm-16color support) introduced with ncurses are not necessarily available.

b) ncurses (new curses)

The updated library available on Linux-like systems. (including the Cygwin plug-in for Microsoft Windows). It provides a platform-independent Application Programming Interface (API) that enables the same application source code to work with proprietary hardware.

7 Python Interpreter / Python Virtual Machine

One or more cross-platform, industry-standard Python programing languages and associated Interpreter and Virtual Machine.

The *Team*STARS "tsWxGTUI_PyVx" Toolkit is implementation in both the mature Python 2x and evolving Python 3x interpreted programming languages.

a) It is precompiled for the platform's processor and operating system by either the operating system or "Cygwin" plug-in manufacturer.

b) It automatically compiles source code into platform independant byte-code during "site-package" installation or else upon Python application launching.

8 Optional Computer Terminal Display for Software Engineering Workstation

Any one of the following may be substituted for the **Basic Computer Terminal Display** based on your need to simultaneously view multiple software engineering documents and activities.

Derived From Wikipedia, the free encyclopedia at "https://en.wikipedia.org/wiki/Display_resolution"

Acronym (usage)	Aspect ratio	Width (pixels)	Height (pixels)
VGA (12" CRT)	4:3	640	480
SVGA (14" CRT)	4:3	800	600
XGA (15" Laptop)	4:3	1024	768
XGA+	4:3	1152	864
WXGA	16:9	1280	720
WXGA	5:3	1280	768
WXGA	16:10	1280	800

SXGA– (UVGA)	4:3	1280	960
SXGA	5:4	1280	1024
HD	~16:9	1360	768
HD	~16:9	1366	768
SXGA+	4:3	1400	1050
WXGA+ (17" Laptop)	16:10	1440	900
HD+	16:9	1600	900
UXGA	4:3	1600	1200
WSXGA+	16:10	1680	1050
FHD	16:9	1920	1080
WUXGA	16:10	1920	1200
QWXGA	16:9	2048	1152
WQHD (27" Desktop)	16:9	2560	1440
WQXGA	16:10	2560	1600

1.1.2 What is the Toolkit designed do?

The *Team*STARS "tsWxGTUI_PyVx" Toolkit's cross-platform design facilitates the creation, enhancement, troubleshooting, maintenance, porting and support of:

1. Mission-critical equipment used by commercial, industrial, medical and military customers. Such equipment can include a broad range of embedded computer systems which satisfy the ***Platform Hardware and Software Requirements*** (on page 3).
2. Application programs used for the local and remote supervisory control and data acquisition associated with automation, communication, control, diagnostic, instrumentation and simulation. Such application software typically includes "operator-friendly" user interfaces.

1.1.2.1 Command Line User Interface (CLI)

Application programs may support none, any, or all of these three major types of command line interface mechanisms:

1. Parameters

Most operating systems support a means to pass additional information to a program when it is launched. When a program is launched from an Operating System command line shell, additional text provided along with the program name is passed to the launched program.

The Toolkit's POSIX-compatible Command Line User Interface (CLI) supports use of:

a) key-word/value pair options

   Example: python -m MODULE_NAME

b) positional arguments

   Example: diff ./Python2xVersion/tsLogger.py ./Python3xVersion/tsLogger.py

2 Interactive command line sessions

After launch, a program may provide an operator with an independent means to enter commands in the form of text.

3 OS inter-process communication

Most operating systems support means of inter-process communication (for example; standard streams or named pipes). Command lines from client processes may be redirected to a CLI program by one of these methods.

Example: python tsLinesOfCodeProjectMetrics.py 2> error.log

1.1.2.2 Graphical User Interface (GUI)

The Toolkit's character-mode emulation of the pixel-mode "wxPython" Graphical User Interface (GUI) creates displays on terminals and terminal emulators with:

1 8-/16-Color (xterm-family) and non-color (vt100-family) Display output

2 Keyboard input

3 Pointer input (mouse, trackball, touchpad and touch screen)

The displays may include horizontal and vertical lines, side-by-side and overlapping windows with or without titles, menu bars, scroll bars, status bars, task bars, buttons, checkboxes, radio boxes & buttons, gauges, date and time stamped event messages, text with or without color and intensity markup, and other GUI objects. Displays may be laid out with or without the help of box and grid sizer services.

1.1.3 What is included in the release?

1.1.3.1 Multi-Project Release

Unlike other "GitHub" repositories which contain a single project, the one for the *Team*STARS "tsWxGTUI_PyVx" Toolkit is organized into four collections of project-specific computer program source code files that the Toolkit recipient will need to install, operate, modify, port and re-distribute the Toolkit.

1 Site-Packages

These two projects are intended to be installed as a registered Python site-package (one for the mature Python 2x programming language and the other for the evolving Python 3x programming language).

Local or remote applications that have imported the appropriate Python 2x or Python 3x "site-package" can be launched from any convenient directory on the associated local or remote computer system.

Modifying a copy of one of these is the most direct way to port the Toolkit to a currently unsupported Python 1x, 2x or 3x platform.

a) Python-2x ("tsWxGTUI_Py2x")

b) Python-3x ("tsWxGTUI_Py3x")

2 Developer-Sandboxes

These two projects are NOT intended to be installed as a registered Python site-package (one for the mature Python 2x programming language and the other for the evolving Python 3x programming language).

Local or remote Python 2x or Python 3x applications can only be launched from the associated "tsWxGTUI_Py2x" or "tsWxGTUI_Py3x" developer-sandbox directory.

Modifying a copy of one of these is the least painful way to experiment with alternative software architectures and algorithms.

a) Python-2x ("tsWxGTUI_Py2x")

b) Python-3x ("tsWxGTUI_Py3x")

1.1.3.2 Consistant User Interface

The four *Team*STARS "tsWxGTUI_PyVx" Toolkit projects are released as a set so that (despite their Python 2x and Python 3x implementation differences) they retain the identical Application Programming Interface (API) and look & feel of their User Interfaces (UI):

1 ***Command Line User Interface (CLI)*** (on page 6)

2 ***Graphical User Interface (GUI)*** (on page 7)

1.1.3.3 Toolkit Components

The *Team*STARS "tsWxGTUI_PyVx" Toolkit include the following components:

1 Documents

The directory contains a collection of files which provide the Toolkit recipient with an understanding of the purpose, goals (capabilities), non-goals (limitations), terms & conditions and procedures for installing, operating, modifying and redistributing the Toolkit.

2 Manual Pages

The directory contains a collection of files which provide a form of online software documentation usually found on a Linux or Unix-like operating systems.

Topics covered include computer programs (library and system calls), formal standards and conventions, and even abstract concepts.

3 Notebooks

The directory contains a collection of files which provide commentaries that express opinions or offerings of explanations about events or situations that might be useful to Toolkit installers, developers, operators, troubleshooters and distributors.

4 SourceDistributions

The directory contains a collection of source code files organized by category:

a) Python programming language (Python 2x and Python 3x)

b) Operating Mode (Command Line Interface and Graphical User Interface)

c) Function (Building Block Libraries, Tools, Tests, Utilities and Examples)

d) Installation (registered Python "Site-Package" and non-registered Python "Developer-Sandbox")

1.1.4 How might you use the Toolkit?

The *Team*STARS "tsWxGTUI_PyVx" Toolkit can save you time by eliminating the need to re-invent, organize and integrate a collection of general purpose, re-usable software building block libraries, tools, tests, utilities and examples.

Here are a few usage situations and associated benefits:

1 Port your existing "curses" software:

Adapt your software so that an executable program can be created for computing environments that are different from the one(s) for which it was originally designed (e.g. different CPU, operating system, or third party library).

You should consider porting when the cost of porting it to a new platform is less than the cost of writing it from scratch. The lower the cost of porting software, relative to its implementation cost, the more portable it is said to be.

a) Existing GUI applications implemented in the low level "c" programming language which use the low level, character-mode "curses" terminal device interface library Application Programming Interface (API) are substantially more costly (in effort) to develop, maintain and enhance than those implemented in higher level "Python" with its somewhat higher level "curses" API.

b) Existing GUI applications implemented in the high level "Python" programming language which use the character-mode "curses"-based emulation of the high level pixel-mode "wxPython" API are substantially less costly (in effort) to develop, maintain and enhance.

2 Port your existing "wxPython" or "wxWidgets" software:

a) Existing GUI applications implemented in the high level "Python" programming language which use the high level pixel-mode "wxPython" API will become portable after removal of those API activities involving icons and curved shapes.

b) Existing GUI applications implemented in the higher level "c++" programming language which use the high level, pixel-mode "wxWidgets" API will become portable after porting to "wxPython and removal of those API activities involving icons and curved shapes.

3 Adapt existing Toolkit software:

a) Find an application among the Toolkit's various CLI and GUI tools, tests and tutorial examples having the structure and user interface closest to your needs.

b) Modify a copy of the Toolkit application to suit your needs.

4 Create new software from scratch:

1.1.5 Where to get further information?

Additional information is available:

1. Learn more about the*Team*STARS "tsWxGTUI_PyVx" Toolkit by browsing its collection of documents, manpages, notebooks (engineering) and source code (building block libraries, tools, tests and examples) at:

   https://github.com/rigordo959/tsWxGTUI_PyVx_Repository

2. Get your own copy of the Toolkit repository (including its records of comments, issue tracking and revisions) via:

   git clone https://github.com/rigordo959/tsWxGTUI_PyVx-Repository