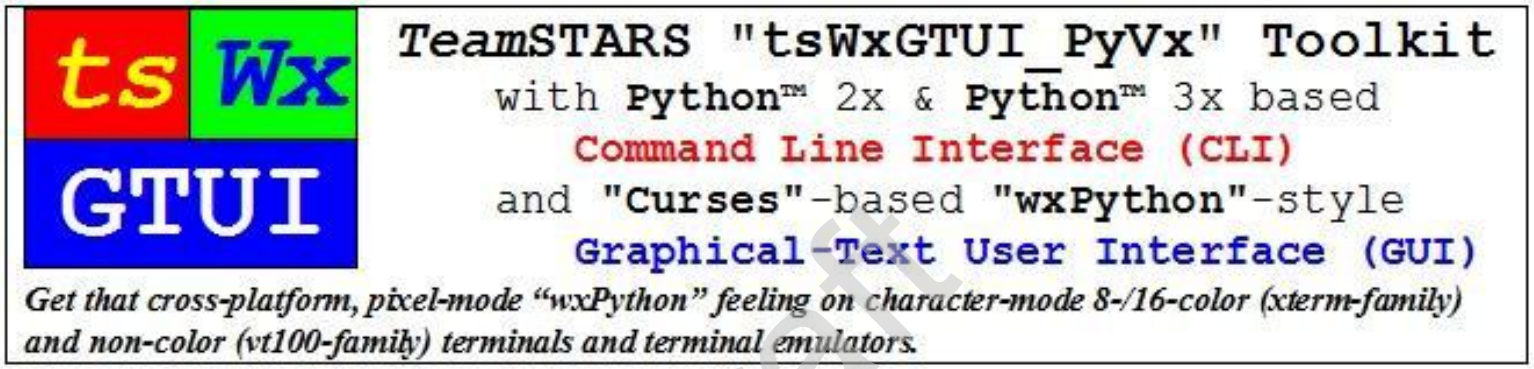



by Richard S. Gordon, a.k.a. Software Gadgetry

# *Table of Contents*


- **DESIGN OVERVIEW**
  - Purpose
  - Application Programs
  - Development Systems
  - Embedded Systems
  - Command Line Interface (CLI)
  - Graphical User Interface (GUI)
- **USAGE OVERVIEW**
  - Capabilities
  - Limitations

# Purpose

1. Provide a framework for creating, enhancing, troubleshooting, maintaining and supporting application programs that are suitable for embedded systems
2. Provide a software development toolkit with:
   - Libraries of building-block components
   - Tools for tracking and improving developer productivity
   - Utilities for monitoring and changing hardware and software configuration
   - Tests (unit, integration, system, regression and demonstration) for design verification and quality assurance

# Application Programs

1. Automation, communication, control, diagnostic, instrumentation and simulation application.
2. Typically require an "operator-friendly" Command Line Interface (CLI) or a Graphical-style User Interface (GUI) that can be controlled locally and/or remotely.

# Development Systems

1. Commercial laptop, desktop, mainframe or super computer systems for development, archiving, maintenance, troubleshooting and publishing of documents and source code.
2. General purpose use with suitable and/or upgradeable processing, memory, communication, input/output and file storage resources.
3. Typically have pixel-mode hardware suitable for email, web surfing, office suite and software development applications using their operating system's graphical user interface.

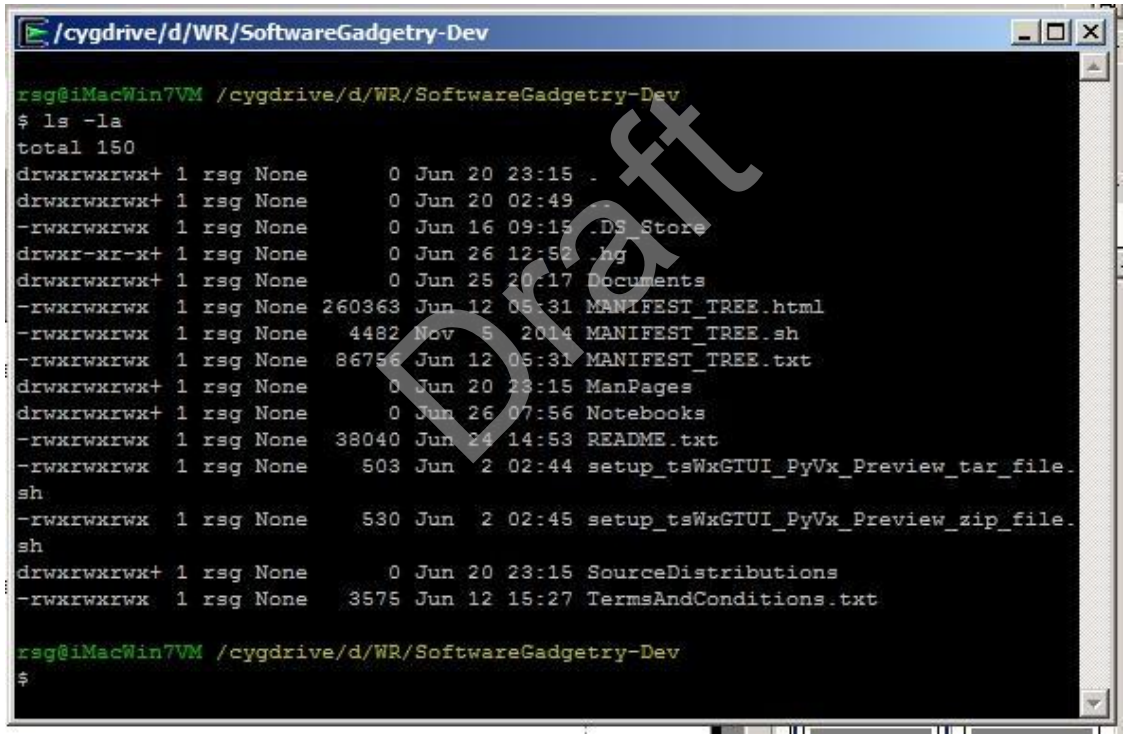
# Embedded Systems

1. Mission-critical systems for commercial, industrial, medical and military applications.
2. Typically customized and optimized for a specific use with limited, application-specific processing, memory, communication, input/output and file storage resources.
3. Typically have character-mode hardware suitable only for troubleshooting and their operating system's command line console.

# Command Line Interface (CLI)

1. Output to the user of a chronological sequence of lines of text written from top to bottom and then scrolling off the top as each new line is written to the bottom of the terminal display.
2. Input from the user via a computer terminal keyboard that is echoed to the display below the previous output.

# Sample CLI Display



# CLI Building Blocks (tsLibCLI)

- Application Building Blocks
  - tsCommandLineInterface
  - tsDoubleLinkedList
  - tsOperatorSettingsParser
  - tsReportUtilities
- Application Diagnostics
  - tsExceptions
  - tsLogger
- Application Configuration
  - tsCxGlobals
  - tsGistGetTerminalSize
  - tsPlatformRunTimeEnviroment
- Application Launchers
  - tsApplication
  - tsCommandLineEnv
  - tsSysCommands

# CLI Tools (tsToolsCLI)

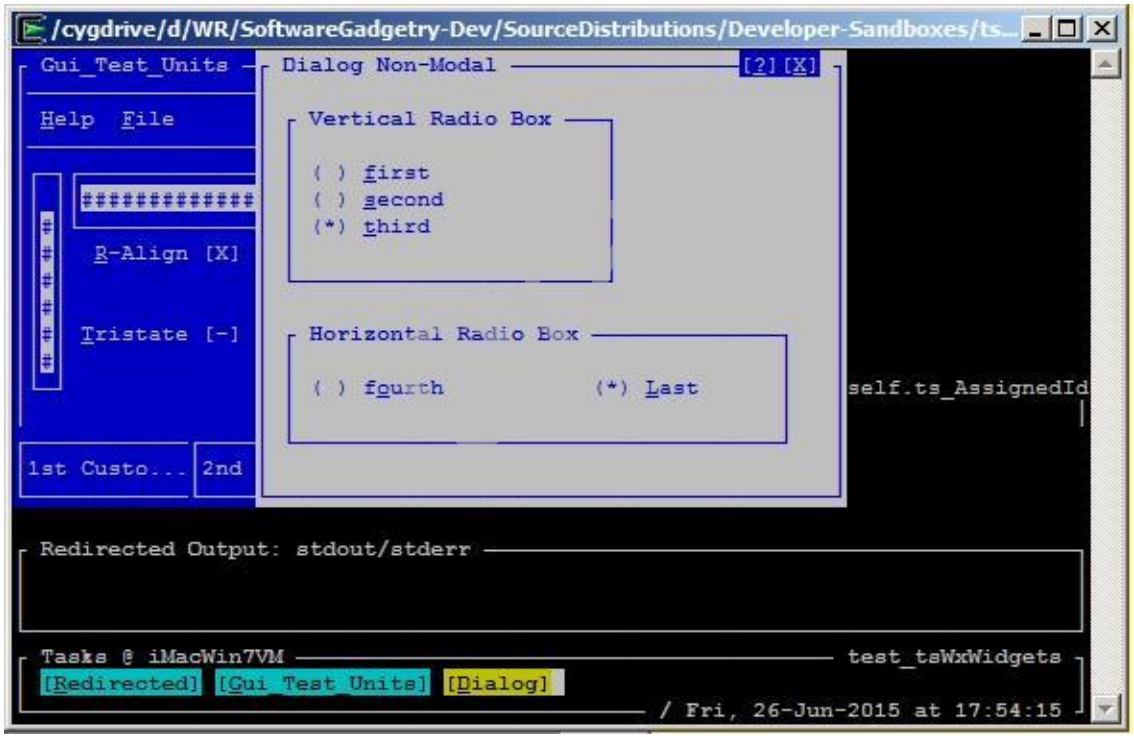
- **Developer Tools**
  - tsChange_tsWxGTUI_PyVx
  - tsChange_wxPython_API_Version
  - tsStripComments
  - tsStripLineNumbers
  - tsTreeCopy
  - tsTreeTrimLines.py
- **Troubleshooter Tools**
  - tsPlatformQuery

- **Developer Productivity Tracking Tools**
  - tsLinesOfCodeProjectMetrics
    - Total Files, Lines of Code & Comments
    - Subtotals (by language) Files, Lines of Code & Comments
    - Estimates labor cost/value, staffing and schedule per COCOMO-81 model. It computes software development effort as a function of program size and a set of "cost drivers" that include subjective assessments of product, hardware, personnel, and project attributes.

# Graphical User Interface (GUI)

1. Output to the user of character strings to application-specified column and row (line) fields on a computer terminal display.
2. Input from the user via a pointing device (such as mouse, trackball, touchpad or touch screen) that is moved into a position by its operator before a mouse button is clicked to activate a function button, radio button , checkbox or keyboard input position.
3. Input from the user via a computer terminal keyboard that is echoed as output to a reserved area of the display that is written from top to bottom and then scrolling off the top as each new line is written to the bottom of the reserved area.

# Sample GUI Display



# GUI Building Blocks (tsLibGUI)

- Application Building Blocks (partial listing)
  - Top-Level GUI Objects
    - tsWxFrame
    - tsWxDialog
  - Lower-Level GUI Objects
    - tsWxPanel
    - tsWxStatusBar
  - GUI Controls
    - wxWxButton
    - tsWxCheckbox
    - tsWxGauge
    - tsWxMenuBar
    - tsWxRadioButton
    - tsWxScrollBar
    - tsWxTaskBar
  - GUI Events
    - Keyboard / Mouse / Tmer
  - GUI Sizers
    - tsWxBoxSizer
    - tsWxGridSizer
- Application Configuration
  - tsWxGlobals
  - tsWxGraphicalTextUserInterface
  - tsWxSplashScreen
- Application Diagnostics
  - tsWxLog (Future)
- Application Launchers
  - tsWxApp
  - tsWxPyApp
  - tsWxPyOnDemandOutputWindow
  - tsWxPySimpleApp
  - tsWxMultiFrameEnv

# GUI Building Blocks (tsLibGUI)

- Application Building Blocks
  - tsCommandLineInterface
  - tsDoubleLinkedList
  - tsOperatorSettingsParser
  - tsReportUtilities
- Application Diagnostics
  - tsExceptions
  - tsLogger
- Application Configuration
  - tsCxGlobals
  - tsGistGetTerminalSize
  - tsPlatformRunTimeEnviroment
- Application Launchers
  - tsApplication
  - tsCommandLineEnv
  - tsSysCommands

# Capabilities: Local Monitoring / Control

1. Once you've logged into your local computer, you can launch one or more local shells (such as "sh" and "bash") associated with the local operating system's command line interface.
2. The local operating system's command line interface provides access to associated terminal interface and the TeamSTARS "tsWxGTUI_PyVx" Toolkit's Python and "nCurses" based character-mode user interfaces which enable you to monitor and control one or more local application programs.

# Capabilities: Remote Monitoring / Control

1. Once you've logged into your local computer, you may then login to a remote computer using one or more secure shells ("ssh") or non-secure shells ("rsh") provided by the local operating system.
2. The Secure Shell ("ssh") is a cryptographic network protocol for secure data communication, remote command-line login, remote command execution, and other secure network services between two networked computers. It connects, via a secure channel over an insecure network, a server and a client running "ssh" server and "ssh" client programs, respectively.
3. The remote shell ("rsh") is a command line computer program that can execute shell commands as another user, and on another computer across a computer network. The remote system to which "rsh" connects runs the "rsh" daemon ("rshd").
4. The local and remote operating system's command line interfaces provides access to associated terminal interface and the TeamSTARS "tsWxGTUI_PyVx" Toolkit's Python and "nCurses" based character-mode user interfaces which then communicate.
5. This enables you to monitor and control one or more remote and local application programs, from the convenience of your local computer terminal, with greater speed and efficiency than possible with the larger communication traffic associated with pixel-mode.

# Limitations: Not Magical Cross-platform

1. Host Specific Native Applications
   - You will NOT be able to run native Linux, Mac OS X, Microsoft Windows and Unix applications on each other's development and embedded system platforms.
2. Pixel-mode "wxPython" / "wxWidgets" Applications
   - You will NOT be able to run pixel-mode applications on platforms with character-mode terminals or terminal emulators (such as vt100, vt220, Cygwin, linux, xterm, xterm-color, xterm-16color, xterm-88color and xterm-256color).