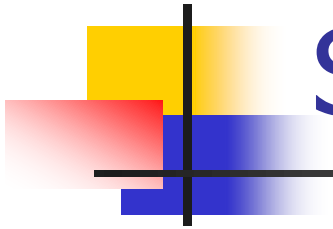
UseCase_7_Source_Distributions

TeamSTARS "tsWxGTUI_PyVx" Toolkit
with **Python** 2x & **Python** 3x based
Command Line Interface (CLI)
and **"Curses"**-based **"wxPython"**-style
Graphical-Text User Interface (GUI)

Get that cross-platform, pixel-mode "wxPython" feeling on platforms with:

- 64-bit processors, nCurses 6.x, 64-bit Python 3.6.x or later GUI applications and character-mode 256-/16-/8- color (xterm-family) and non-color (vt100-family) terminals and terminal emulators.
- 32-bit processors, nCurses 6.x/5.x, 32-bit Python 3.5.2 or earlier GUI applications and character-mode 16-/8-color (xterm-family) and non-color (vt100-family) terminals and terminal emulators.

Table of Contents (*with slide show* Hyperlinks)



Source Code

- Is any collection of computer instructions (possibly with comments) written using some human-readable computer language, usually as text.
- Is specially designed to facilitate the work of computer programmers, who specify the actions to be performed by a computer.
- Is often transformed by a compiler program into low-level machine code understood by the computer. The machine code might then be stored for execution at a later time.
- Alternatively, an interpreter (such as the ones for the Python 2x and Python 3x languages) can be used to analyze and perform the outcomes of the source code program directly on the fly.

Source Code Organization

- Small, simple computer programs may be contained within a single file.
- To facilitate development, testing and maintenance, larger, more complex computer programs are typically subdivided into smaller files within a common directory.
- General purpose, re-usable files are grouped together, by commonality of function and interface, into a building-block library directory.
- A larger, more complex system may organize building-block libraries, tests, examples and tutorials into separate directories and/or subdirectories.

Site-Packages (full listing) (Table of Contents)

- Site-Packages are the location where third-party packages are installed (i.e., those not part of the core Python distribution).
  - A Site-Package can only be used with those Python versions for which it has been explicitly installed.
  - For Linux, Mac OS X and Unix operating systems, one must have "root" or administrator privileges to write to the install location.
- To facilitate application software development and deployment, Site-Packages:
  - Organize source code into a single layer set of subdirectories within the Site-Package directory.
  - Import modules, via static Site-Package directory relative path specifications.
- **Default Python Example**: A Site-Package, installed for use with the platform's default Python must be installed via
  - "python setup.py install"
- **Optional Python 2x Example**: A Site-Package, installed for use with the platform's add-on Python 2.7.10 must be installed via
  - "python2.7.10 setup.py install"
- **Optional Python 3x Example**: A Site-Package, installed for use with the platform's add-on Python 3.5.0 must be installed via
  - "python3.5.0 setup.py install

Developer-Sandboxes (full listing)

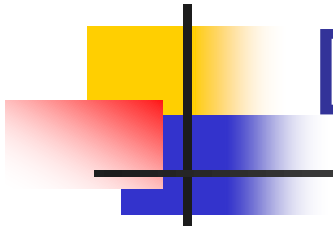
(Table of Contents)

- A Developer-Sandbox is a testing environment that isolates untested code changes and outright experimentation from the production (Site-Package) environment or repository.
  - There should be one Developer-Sandbox for any or all Python 2x (mature, second generation language) releases.
  - There should be one Developer-Sandbox for any or all Python 3x (evolving, third generation language) releases.
- To facilitate software development and troubleshooting, Developer-Sandboxes:
  - Organize source code into a set of multi-level nested directories within the Developer-Sandbox directory.
  - Import modules, within try-except blocks, via dynamic multi-level nested path specifications.

Directory Structure (example):

```
tsLibCLI/ # Package for imported library building blocks and associated non-imported test applications

  __init__.py
    |
  tsApplicationPkg
    |   |
    |   +-- src (Subpackage for imported building blocks)
    |   |     |
    |   |     +-- __init__.py
    |   |     |
    |   |     +-- tsApplication.py
    |   |
    |   +-- test (Container for non-imported test applications)
    |         |
    |         +-- test_tsApplication.py
    |
  tsCxGlobalsPkg
    |
  tsExceptionPkg
    |
  tsPlatformRunTimeEnvironmentPkg
    |
  tsReportUtilityPkg
```



Dynamic Import (example)

Source Code, Tests, Tools and Utilities for users of the second generation language, Python 2.0.0-2.7.9.
This version uses a dynamic import mechanism, suitable for nested multi-level packages, in which each application and building block module:

a) imports required library packages simply by name. Examples:

```
import tsLibCLI
import tsLibGUI
```

b) imports required building block modules, and defines optional aliases, simply by name. Examples:

```
import tsCxGlobals
import tsExceptions as tse
import tsWx as wx
from tsReportUtilities import TsReportUtilities as tsrpu
```

c) Library package "__init__.py" modules dynamically construct any associated full paths.