

Contents

- [1 Script Shell](#)
- [2 Python core](#)
- [3 Compatibility](#)
- [4 Standard modules](#)
- [5 3rd Party Extensions](#)
- [6 Packaging \(SIS\)](#)
- [7 How does SIS work](#)

Script Shell

- On device lists files in c:\python e:\python c:\data\python e:\data\python
- Emulator lists only c:\data\python

Python core

- 1.4.5 uses python 2.2.2 ([Python 2.2.2 Documentation](#))
- 1.9.5 uses **python 2.5.4** ([Python 2.5.4 Documentation](#))

Compatibility

- **Compatibility break between 1.9.3 and 1.9.4**, since most of the standard python modules and the extension modules are moved from runtime package to Module repository. Just repackage your application using 1.9.4 application packager.
- Starting from PyS60 1.9.5, **e32.s60_version_info** returns the actual S60 platform version of the device, not the SDK version on which PyS60 was compiled.

Standard modules

- import is case-sensitive: topwindow is different than TopWindow
- on 1.9.x **socket** and **calendar** refer to the Python core modules (vs S60 versions)
- socket vs btsocket
- calendar vs e32calendar

3rd Party Extensions

- **Extensions made for 1.4.x have to be built again for 1.9.x**
- Instructions in zip
- Make a ZIP that users can unpack into the modulerepository directory. That ZIP should contain extension's .pyd's and .py's as well as the dependency specification file that says which modules it imports from the standard library. Then the packaging tool can simply embed the necessary extension files into the application as it's packaged.
- At the time of creating SIS the application packager (ensymbler) automatically scans your python script, finds out all the imports, checks if those modules are part of Python runtime. If not it will package them with your application and they will be installed on the phone at the standard location(\sys\bin\).
- If you are developing a library or application which other developers will use (miso, wlantools, pygame etc..) you should create a zip package in the specified format (please refer doc) and then upload it. Other developers will download this zip, extract it to the module repo on their PC. When they package their script which imports this extension, the application packager will search for this module in the module repo and then package the PYD/PY along with the application.
- How are conflicts handled, when 2+ applications need same extension? The PYD names are suffixed with the application UID during packaging.

Packaging (SIS)

- PyS60 Application Packager (ensymbler with graphical UI)
- Command line usage: copy folders **module-repo** and **templates** into directory where you run ensymbler

How does SIS work

- python_ui.exe is the S60 UI stub app which creates the UI environment and loads appuifw. At the end it runs the application's default.py