PEP 3147: PYC cache directory

Neil Muller

July 17, 2010

The Problem

- Compiled versions of python modules (pyc) written when possible to save import time later (A good thing)
- PYC files are specific to python major versions (so?)
- What happens when we use multiple python versions on the same module?

The Problem

- Compiled versions of python modules (pyc) written when possible to save import time later (A good thing)
- PYC files are specific to python major versions (so?)
- What happens when we use multiple python versions on the same module?
- Additional complication - distributions want to support multiple python versions
  - Want to install/created compiled versions of system modules
  - But don't want to have multiple packages per python version (unmaintainable) or multiple copies of the python files (wastes space)
- Current solutions
  - Oh noes, too hard, only support 1 python version (some Fedora versions)
  - symlink farm - messy, hard to maintain (Ubuntu, Debian)

The Solution

- Create dedicated directory for pyc files __pycache__
- pyc files are annotated with version and implementation (magic number used in pyc file)
    - i.e. .cpython-32.pyc , etc.
- Allows for single installation for multiple python versions and multiple python implementations

The Solution

- Create dedicated directory for pyc files __pycache__
- pyc files are annotated with version and implementation (magic number used in pyc file)
  - i.e. .cpython-32.pyc , etc.
- Allows for single installation for multiple python versions and multiple python implementations
- Special tweaks
  - module.__file__ now refers to original py file
  - new __cached__ attribute refers to pyc file (although details of __cached__ are implementation specific)
  - pyc files only loaded from __pycache__ if py file exists where excepted
- legacy pyc file handling
  - pyc next to source file ignored if source file exists
  - pyc not in __pycache__ will be imported if no source file
    - i.e. pyc only distribution still possible (<insert favourite editorial rant about sourceless code distribution here>)

Future Directions

- Version embedded in compiled extension modules (PEP-3149)
  - Avoids needing multiple installation directories for each python version supported
  - Much fiddlier problem
    - ABI incompatibilities not strictly tied to version - depends on features used, etc.
    - compilation options (debugging, etc.) often cause ABI incompatibilities for the same version, though
    - Ongoing discussion of requirements for a good solution
    - Interactions with PEP 384 (stable API) an issue
  - Likely to happen

Future Directions

- Version embedded in compiled extension modules (PEP-3149)
    - Avoids needing multiple installation directories for each python version supported
    - Much fiddlier problem
        - ABI incompatibilities not strictly tied to version - depends on features used, etc.
        - compilation options (debugging, etc.) often cause ABI incompatibilities for the same version, though
        - Ongoing discussion of requirements for a good solution
        - Interactions with PEP 384 (stable API) an issue
    - Likely to happen
- Backports to older python versions
    - Won't happen for any official builds
    - Almost certain to happen for distribution builds of python 3.1
    - Likely to happen for Python 2.6 & 2.7 in Ubuntu (although probably disabled by default)