Distributing Python applications with PyInstaller

by Giovanni Bajo - rasky@develer.com

22/6/2011
EuroPython 2011

Giovanni Bajo

. software . hardware . innovation

- 9 yrs Python experience
- Maintainer of PyInstaller
- Organizer of PyCon Italy & EuroPython
- CTO at Develer

@giovannibajo

http://giovanni.bajo.it

develer

Goal

. software . hardware . innovation

Ship applications written in Python

without any dependency

on Win, Mac, Linux

develer

Means

. software . hardware . innovation

No compilation possible

Package and ship it

develer

Solution:

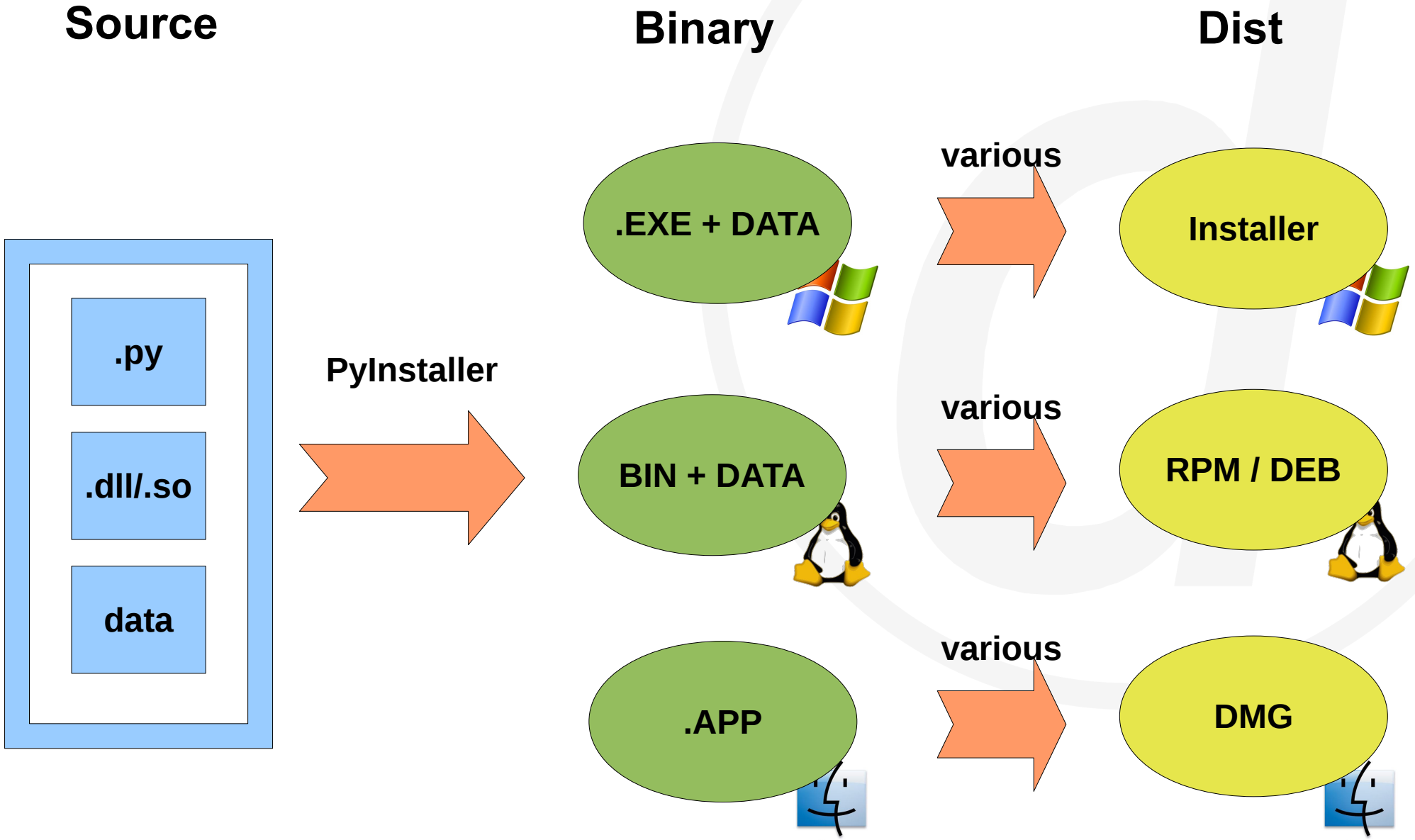
PyInstaller

http://www.pyinstaller.org

develer

Overview

. software . hardware . innovation



develer

Crash tutorial

. software . hardware . innovation

```
#!/usr/bin/env python
"""
Compute the cross spectral density of two signals
"""
import numpy as np
import matplotlib.pyplot as plt

# make a little extra space between the subplots
plt.subplots_adjust(wspace=0.5)

dt = 0.01
t = np.arange(0, 30, dt)
nse1 = np.random.randn(len(t))                 # white noise 1
nse2 = np.random.randn(len(t))                 # white noise 2
r = np.exp(-t/0.05)

cnse1 = np.convolve(nse1, r, mode='same')*dt   # colored noise 1
cnse2 = np.convolve(nse2, r, mode='same')*dt   # colored noise 2

# two signals with a coherent part and a random part
s1 = 0.01*np.sin(2*np.pi*10*t) + cnse1
s2 = 0.01*np.sin(2*np.pi*10*t) + cnse2

plt.subplot(211)
plt.plot(t, s1, 'b-', t, s2, 'g-')
plt.xlim(0,5)
plt.xlabel('time')
plt.ylabel('s1 and s2')
plt.grid(True)

plt.subplot(212)
cxy, f = plt.csd(s1, s2, 256, 1./dt)
plt.ylabel('CSD (db)')
plt.show()
```

- Matplotlib example
- Uses numpy, PyQt4

develer

Crash tutorial

1) Fetch PyInstaller & unpack

Downloads

File	MD5	Description
Release 1.5		
PyInstaller 1.5 (tar.bz2)	e723e703a5b622d556854a1938b8b4c1	Stable Release
PyInstaller 1.5 (zip)	57e3f5c0b0652f43b9a4eb6ff0d4960b	Stable Release
Pyinstaller SVN trunk (zip)		Hourly snapshot

Unpack to writable directory ($PYINST)

develer

Crash tutorial

2) Configure PyInstaller

```
> python $PYINST/Configure.py
```

Crash tutorial

3) Create project

```
> $PYINST/Makespec.py mpl.py
wrote /home/rasky/src/prova/mpl.spec
now run Build.py to build the executable
```

Crash tutorial

4) Build executable

```
> $PYINST/Build.py mpl.spec
```

Crash tutorial

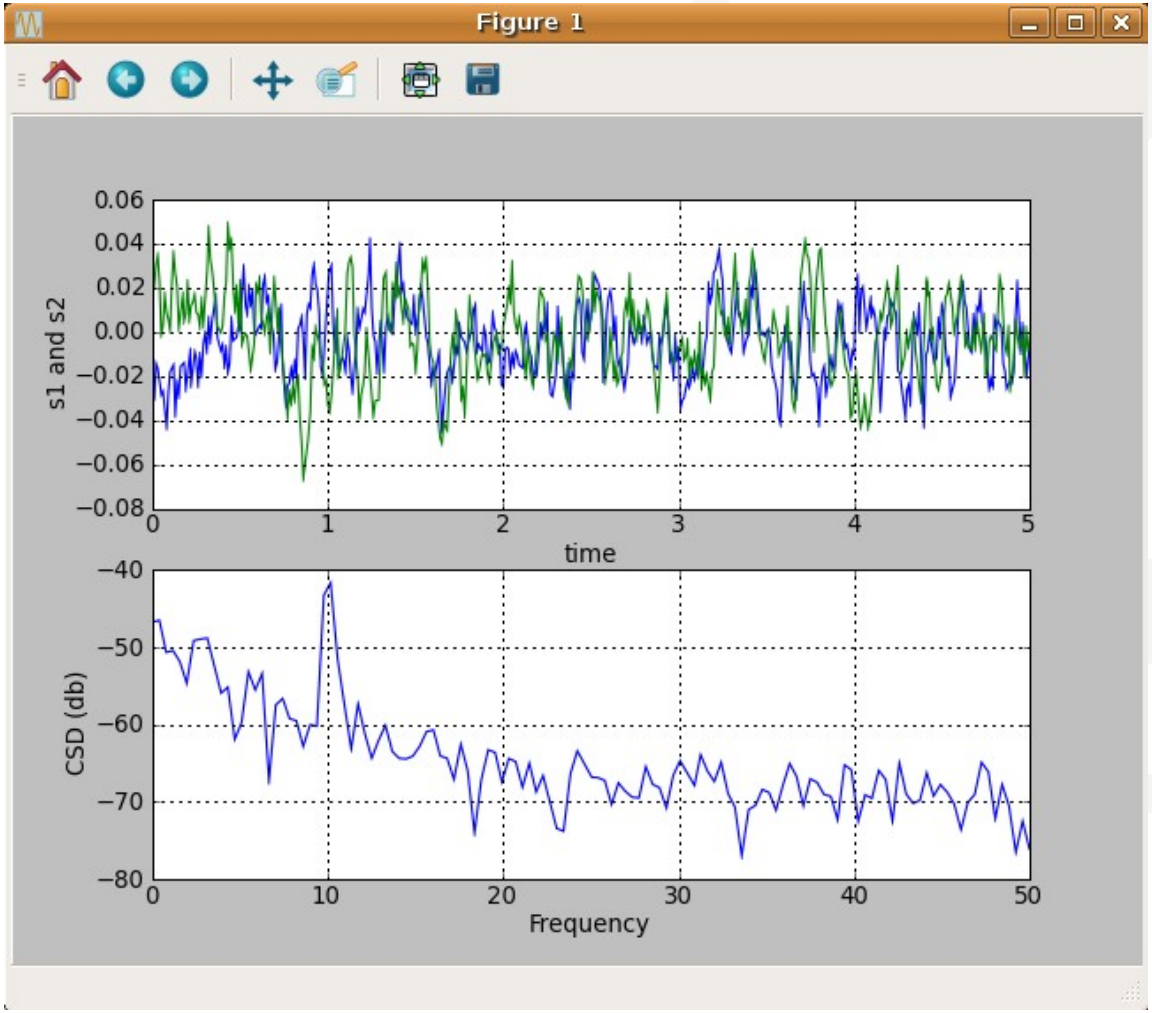
4) Run & Enjoy

```
> cd dist/mpl
> ./mpl
```

Crash tutorial

. software . hardware . innovation

Linux & PyQt

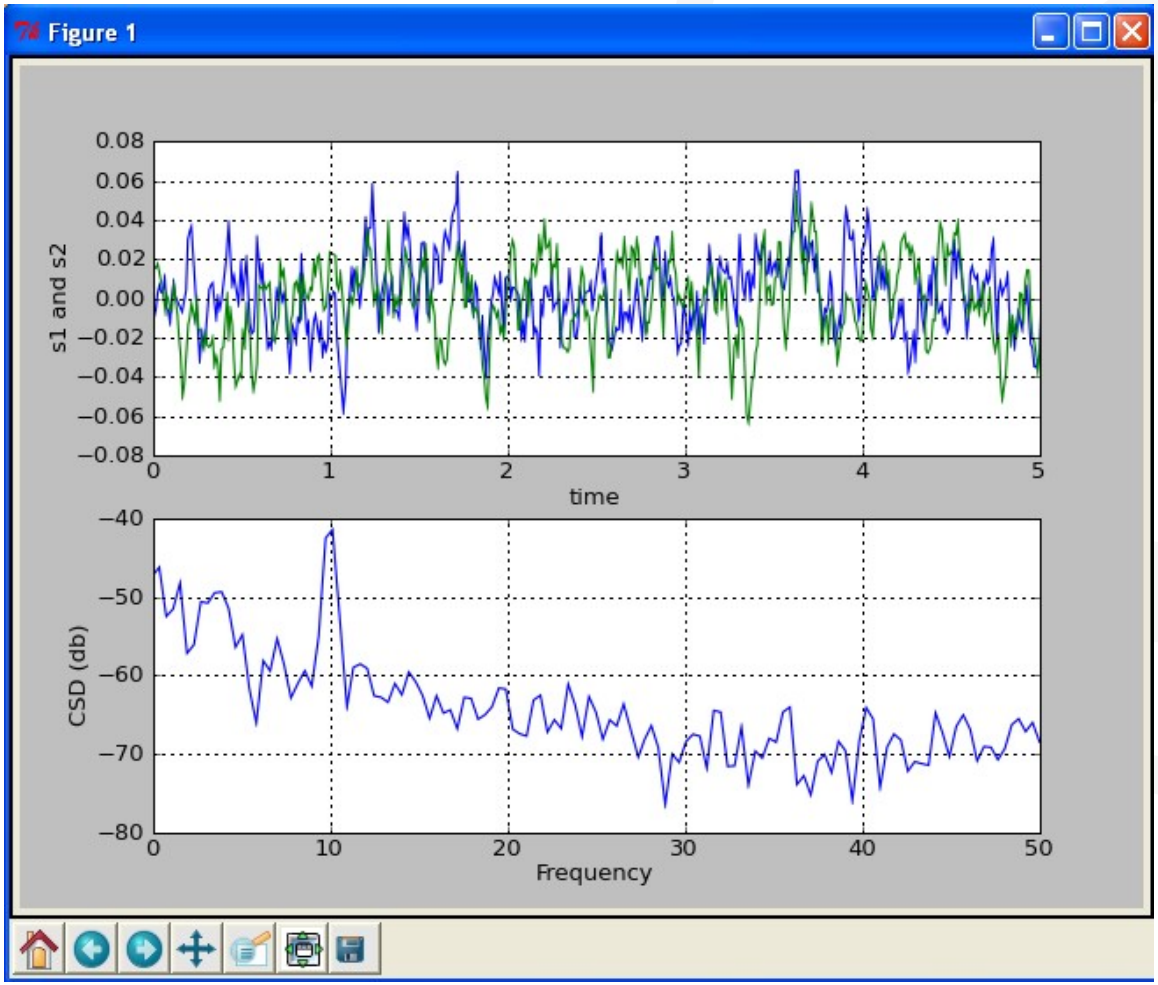


develer

Crash tutorial

. software . hardware . innovation

Windows & Tk

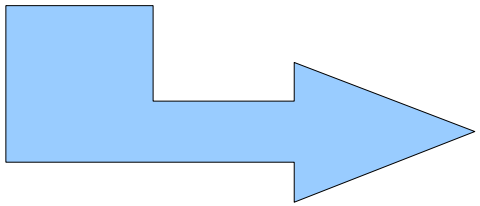


develer

What is in dist/project/?

. software . hardware . innovation

Main executable



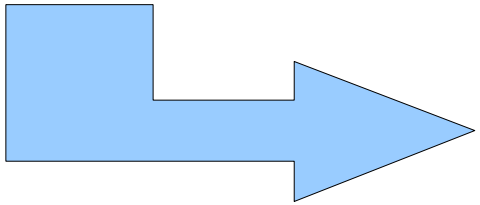
```
total 4,3M
 40K bz2.so
148K _codecs_cn.so
156K _codecs_hk.so
 24K _codecs_iso2022.so
260K _codecs_jp.so
132K _codecs_kr.so
108K _codecs_tw.so
 88K datetime.so
 24K _heapq.so
641K helloworld
2,7M libpython2.6.so.1.0
 36K _multibytecodec.so
 28K readline.so
```

develer

What is in there

. software . hardware . innovation

Python library

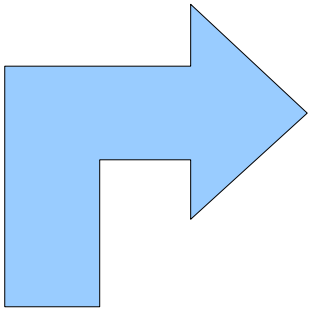


```
total 4,3M
 40K bz2.so
148K _codecs_cn.so
156K _codecs_hk.so
 24K _codecs_iso2022.so
260K _codecs_jp.so
132K _codecs_kr.so
108K _codecs_tw.so
 88K datetime.so
 24K _heapq.so
641K helloworld
2,7M libpython2.6.so.1.0
 36K _multibytecodec.so
 28K readline.so
```

develer

What is in there

. software . hardware . innovation



Standard library

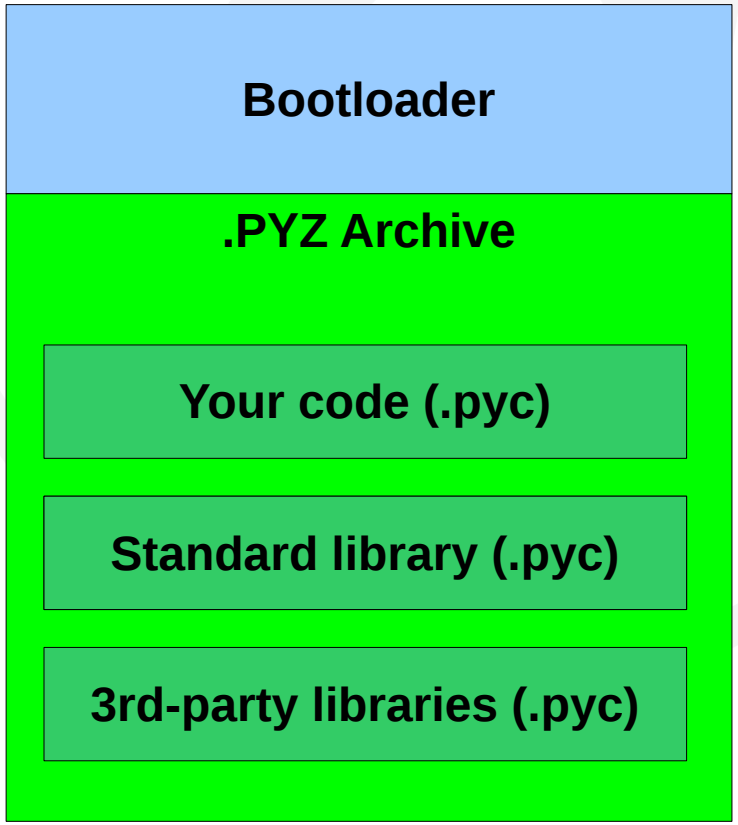
```
total 4,3M
 40K bz2.so
148K _codecs_cn.so
156K _codecs_hk.so
 24K _codecs_iso2022.so
260K _codecs_jp.so
132K _codecs_kr.so
108K _codecs_tw.so
 88K datetime.so
 24K _heapq.so
641K helloworld
2,7M libpython2.6.so.1.0
 36K _multibytecodec.so
 28K readline.so
```

develer

Where is the program?

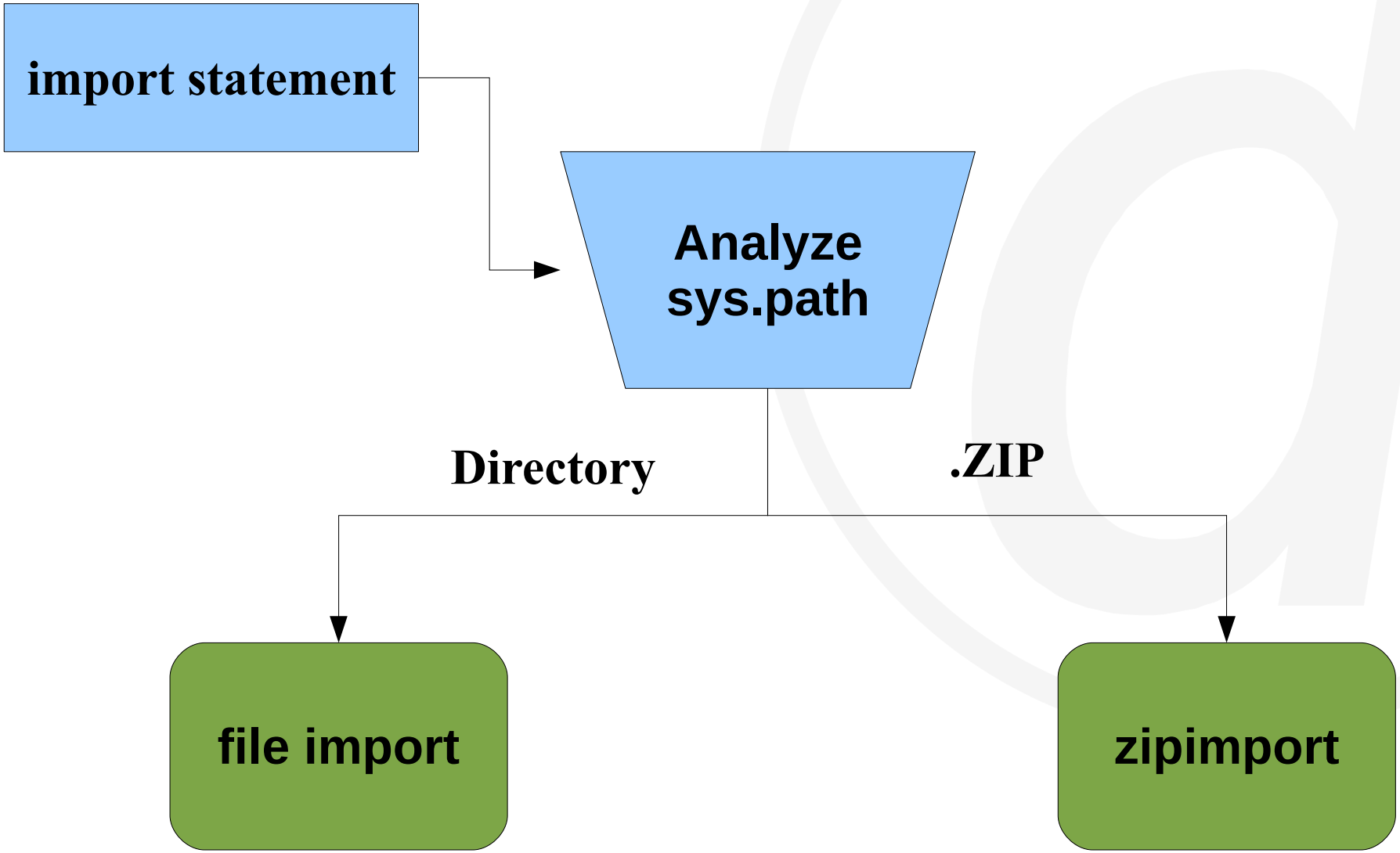
- Automatic discovery
- ZLIB compression
- Never extracted to disk!

helloworld binary:



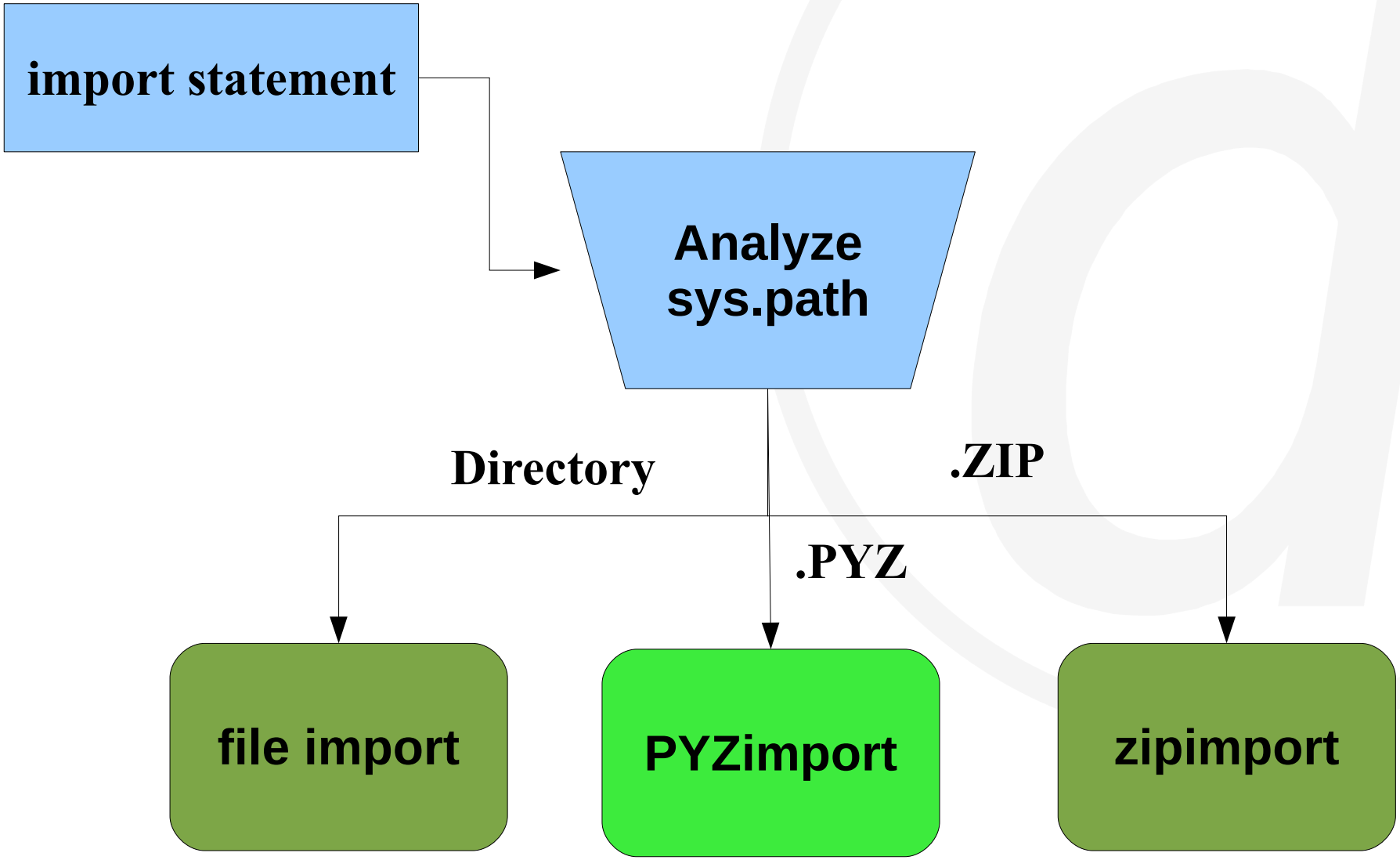
develer

Import hooks



develer

Import hooks



So many files...

Can we do better?

develer

And now for something better

REDUX: 3) Create project in “one-file” mode

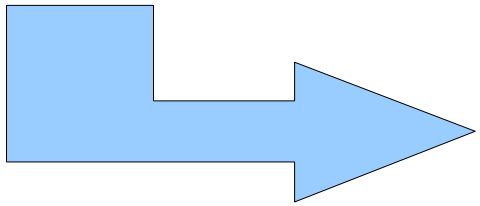
```
> $PYINST/Makespec.py \
    --onefile \
    mpl.py
wrote /home/rasky/src/prova/mpl.spec
now run Build.py to build the executable

> $PYINST/Build.py mpl.spec
```

What is in there?

. software . hardware . innovation

Everything!



```
total 2,1M
2,1M helloworld
```

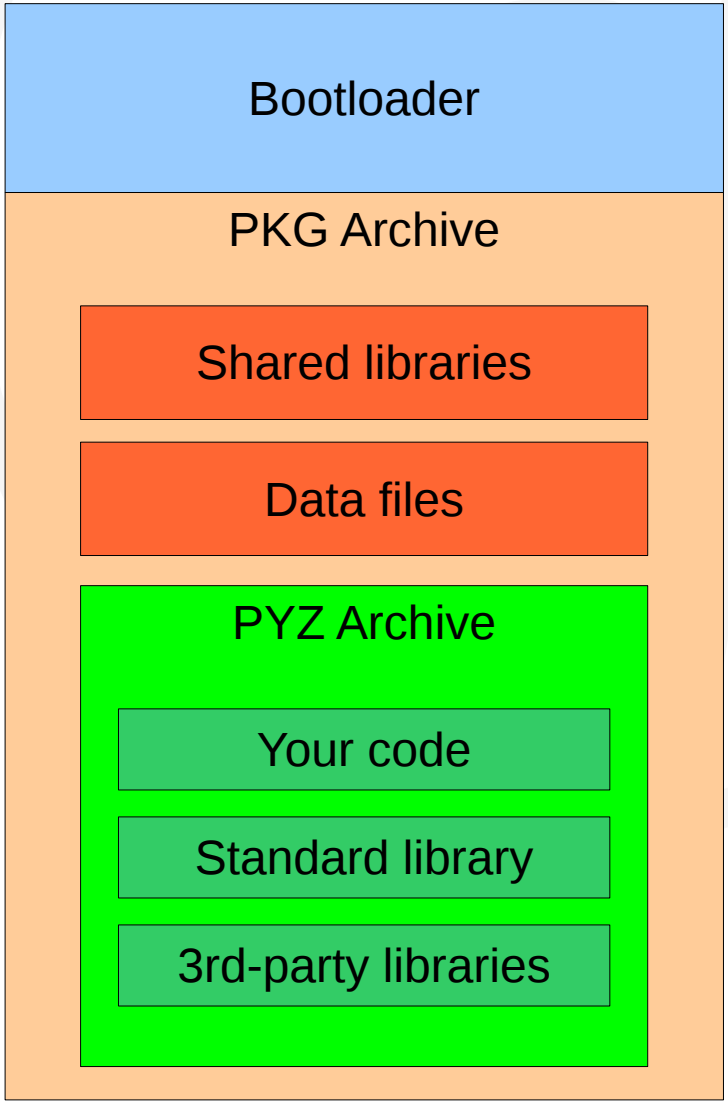
- Compressed package
- No dependencies
- Ready to download & run

develer

Where are the libraries?

Within the executable:

- All compressed
- Extracted to tmp
- Slight overhead
- PYZ within PKG



Explore the executable

. software . hardware . innovation

Use ArchiveViewer.py

> $PYINST/ArchiveViewer.py helloworld

```
 pos, length, uncompressed, iscompressed, type, name
[(0, 558787, 558787, 0, 'z', 'outPYZ1.pyz'),
 (558787, 8146, 20907, 1, 'm', 'iu'),
 (566933, 148, 197, 1, 'm', 'struct'),
 (567081, 5919, 14727, 1, 'm', 'archive'),
 (573000, 1157, 2272, 1, 's', '_mountzlib'),
 (574157, 81, 76, 1, 's', 'useUnicode'),
 (574238, 32, 24, 1, 's', 'helloworld'),
 (574270, 11948, 34952, 1, 'b', '_multibytecodec.so'),
 (586218, 34349, 88616, 1, 'b', 'datetime.so'),
 (620567, 8306, 22672, 1, 'b', '_codecs_iso2022.so'),
 (628873, 62633, 108656, 1, 'b', '_codecs_tw.so'),
 (691506, 92362, 264304, 1, 'b', '_codecs_jp.so'),
 (783868, 7936, 21664, 1, 'b', '_heapq.so'),
 (791804, 100644, 149616, 1, 'b', '_codecs_cn.so'),
 (892448, 75514, 133232, 1, 'b', '_codecs_kr.so'),
 (967962, 8723, 26976, 1, 'b', 'readline.so'),
 (976685, 14432, 38344, 1, 'b', 'bz2.so'),
 (991117, 34586, 157840, 1, 'b', '_codecs_hk.so'),
 (1025703, 1051388, 2741440, 1, 'b', 'libpython2.6.so.1.0')]
?
```

develer

Explore the executable

. software . hardware . innovation

```
? O outPYZ1.pyz
 Name: (ispkg, pos, len)
{'StringIO': (False, 264167, 4586),
 'UserDict': (False, 237735, 2725),
 '__future__': (False, 30473, 1799),
 '_abcoll': (False, 476330, 5942),
 '_strptime': (False, 546070, 6467),
 '_threading_local': (False, 347154, 2569),
 'abc': (False, 158772, 2444),
 'base64': (False, 319016, 4335),
 'bdb': (False, 398051, 6813),
 'bisect': (False, 463350, 956),
 'calendar': (False, 146871, 9221),
 'cmd': (False, 79468, 6039),
 'codecs': (False, 246134, 10207),
 'collections': (False, 50336, 2918),
 'copy': (False, 391940, 4795),
 'copy_reg': (False, 445339, 2460),
 'difflib': (False, 180836, 21728),
```

develer

Feature reel

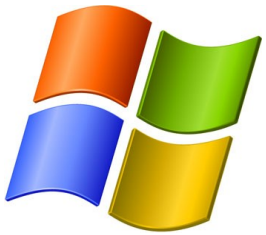
- Free as in beer & freedom
- Multiplatform
- Multiversion (Python 2.0+, no Python 3)
- Flexible packaging mode 
- Advanced built-in support for specific 3rd-party libraries 
- Compression (ZLIB, UPX)

develer

 = PyInstaller exclusive

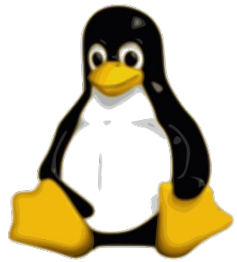
Multiplatform: Windows

- EXEs run on Windows 2000+ **vanilla**
  - No separate CRT/runtime installation!
- Windowed or console mode
- UPX highly compresses DLLs
- Can embed icons
- Can modify PE version infos
- Code-signing fully supported
- COM server support (deprecated)



Multiplatform: Linux

- No fixed ABI
- Ship all libraries (including system ones)
  - Distribution-independent by default
- Can't disable stdout/stderr
- No UPX on .SO



Multiplatform: Mac OS X

- Create bundle (.app) for windowed app
- Forward compatible
  - Build on 10.x, run on 10.x, 10.x+1, etc.
- Can embed icons
- No universal binaries
- No UPX on .so/.dylib

Flexibility

- SPEC file: PyInstaller project (in python)
- Generated by Makespec.py (wizard)
  - One-file / one-dir modes
  - Windowed / console
  - Debug mode
  - Icon, version, etc.

SPEC file


```
# -*- mode: python -*-
a = Analysis(
    [ os.path.join(HOMEPATH,'support/_mountzlib.py'),
      os.path.join(HOMEPATH,'support/useUnicode.py'),
      'helloworld.py' ],
    pathex=['/home/rasky/src/prova'])

pyz = PYZ(a.pure)

exe = EXE( pyz,
           a.scripts,
           a.binaries,
           a.zipfiles,
           a.datas,
           name=os.path.join('dist', 'helloworld'),
           debug=False,
           strip=False,
           upx=False,
           console=1 )
```

Analysis


```
a = Analysis(
    [ os.path.join(HOMEPATH,'support/_mountzlib.py'),
      os.path.join(HOMEPATH,'support/useUnicode.py'),
      'helloworld.py' ],
    pathex=['/home/rasky/src/prova'])
```

- input = bootstrap scripts
- pathex = PYTHONPATH
- Recursively analyze bytecode (.pyc)
- Find out dependencies: Python, OS, mixed

Dependencies: Python level

- Explicit imports
- Conservative on conditional imports:

```
if something:
    import XXXXX
```

- Hidden imports: dynamic generation of name

```
for n in plugins:
    __import__("plugin_" + n)
```

Dependencies: OS level

- Binary dependencies between libraries:
  - Windows: PE import section + manifest parsing
  - Linux: LDD
  - Mac: otool
- Hidden imports: LoadLibrary() / dlopen()
- Try with bindepend.py:

```
> bindepend.py c:\Python25\DLLs\_ctypes.pyd

c:\Python25\DLLs\_ctypes.pyd ['KERNEL32.dll',
'ole32.dll', 'OLEAUT32.dll', 'python25.dll',
'MSVCR71.dll']
```

Dependencies: PY+OS level

- ctypes automatically supported:

```
from ctypes import *

CDLL("library.so")
WinDLL("library.so")

cdll.library
windll.library

cdll.LoadLibrary("library.so")
windll.LoadLibrary("library.so")
```

Hidden imports (& datas)

- Manually maintained list of hidden imports
  - Major 3rd-party packages supported

```
hiddenimports = ['sip', 'PyQt4.QtCore', 'PyQt4._qt']

from hooks.hookutils import qt4_plugins_dir
pdir = qt4_plugins_dir()

datas = [
     (pdir + "/imageformats/*.so",      "qt4_plugins/imageformats"),
     (pdir + "/imageformats/*.dll",     "qt4_plugins/imageformats"),
     (pdir + "/imageformats/*.dylib",   "qt4_plugins/imageformats"),

     (pdir + "/iconengines/*.so",       "qt4_plugins/iconengines"),
     (pdir + "/iconengines/*.dll",      "qt4_plugins/iconengines"),
     (pdir + "/iconengines/*.dylib",    "qt4_plugins/iconengines"),

     (pdir + "/accessible/*.so",        "qt4_plugins/accessible"),
     (pdir + "/accessible/*.dll",       "qt4_plugins/accessible"),
     (pdir + "/accessible/*.dylib",     "qt4_plugins/accessible"),
]
```

SPEC hand-editing

- Hand-edit SPEC file to:
  - Manual includes/excludes
  - Data files
  - Custom packaging mode (eg: “open-source” mode)

Reverse-engineering

- No source code shipped (by default)
- Byte-code easy to extract (ArchiveViewer)
- Basic crypt support in PyInstaller (to be customized)
- Bytecode scrambling still the best option

PyInstaller future

- Simplify basic usage (pyinstaller app.py)
- Improving 3rd-party library support
- Python 3 support
- Cover also the “dist” phase

Questions?

. software . hardware . innovation

develer

Thank you!

Develer S.r.l.
Via Mugellese 1/A
50013 Campi Bisenzio
Firenze - Italia

Contacts

Mail: info@develer.com

Phone: +39-055-3984627

Fax: +39 178 6003614

http://www.develer.com

develer

. software . hardware . innovation