

# Gerando um programa executável a partir de um módulo Python

Wendel Melo (Departamento de Ciência da Computação – UFRJ),  
Notas de aula de Computação II  
15/03/2012

A linguagem de programação Python segue o modelo de interpretação, isto é, exige a presença de um interpretador para a execução das instruções contidas em um programa ou *script*. Esta exigência impossibilita a execução de programas Python em ambientes onde um interpretador da linguagem não esteja disponível. Para usuários Linux, essa questão não chega a ser um problema uma vez que as principais distribuições desse sistema operacional livre já trazem um interpretador Python instalado. No universo Windows, entretanto, o sistema operacional não conta com um interpretador Python vindo de “fábrica”, o que, a princípio, traz limitações para a disseminação dos nossos programas feitos nessa linguagem. Nesse texto, faremos passo-a-passo um pequeno exemplo para gerar um programa executável a partir de um módulo Python para o ambiente Windows, usando o pacote py2exe. O programa executável gerado deverá, em princípio, funcionar em qualquer ambiente Windows que possua as principais bibliotecas compartilhadas instaladas (DLLs).

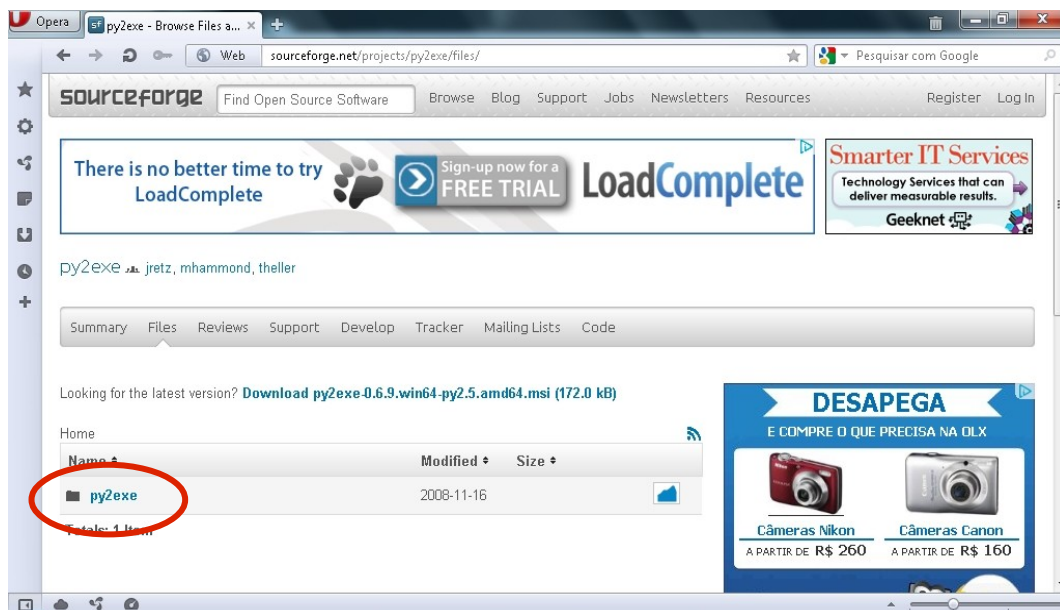
## Instalando o pacote py2exe

O pacote py2exe é uma extensão dos chamados utilitários de distribuição da linguagem Python (distutils) criada para a conversão de módulos Python em programas executáveis Windows. Os programas executáveis gerados são construídos para rodar de forma independente do interpretador Python, o que significa que podem ser utilizados em ambientes onde o mesmo não esteja instalado. Na prática, o que o pacote faz é “encaixotar” no programa executável módulos e funções do interpretador juntamente com seu script. Desse modo, não se deve esperar ganho de desempenho ao rodar um programa Python como executável independente.

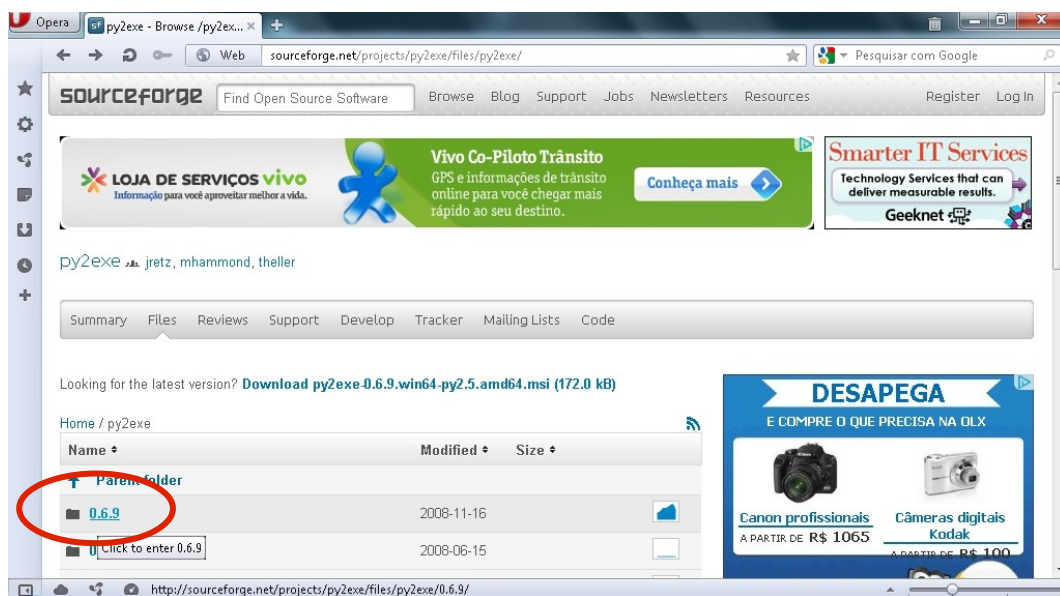
Site oficial: <http://www.py2exe.org/>

Endereço para *download*: <http://sourceforge.net/projects/py2exe/>

É preciso ter um pouco de atenção na hora de baixar o pacote py2exe. Você deve baixar uma versão compatível com sua arquitetura (32 ou 64 bits) e com sua versão do Python. Portanto, você não deve baixar a versão que aparece na capa do site se esta não for adequada ao seu contexto. Em muitos casos, pode ser necessário navegar na estrutura de arquivos do site de download para encontrar a versão apropriada. Para isso, clique, no link 'Files' ou acesse <http://sourceforge.net/projects/py2exe/files/> , conforme a figura a seguir:



Clique na pasta py2exe. Em seguida, escolha uma versão do pacote, preferencialmente a mais recente. No momento em que este texto foi escrito, tal versão era a 0.6.9:



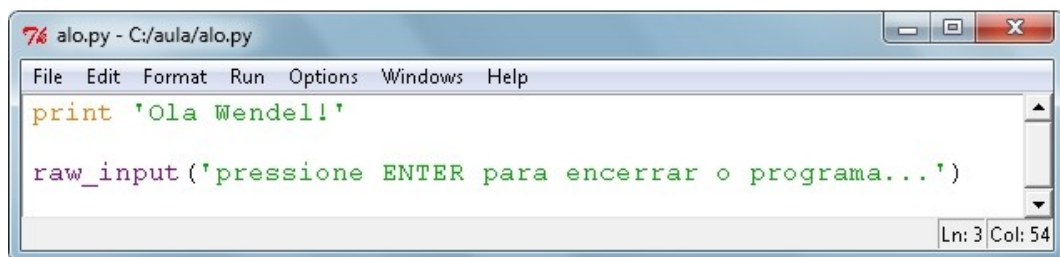
Agora você deve escolher a versão mais adequada para sua arquitetura e versão do Python instalada. Embora minha máquina seja 64 bits, a minha instalação Python é de 32 bits, versão 2.7. Sendo assim, escolherei a versão py2exe para essas características (py2exe-0.6.9.win**32**-py**2.7**.exe). Se você não está seguro se sua instalação Python é de 32 ou de 64 bits, então com 99% de probabilidade a sua instalação é de 32 bits (mais comum). Você pode conferir essa informação olhando a linha de cabeçalho impressa pelo interpretador Python no momento de sua execução:



Após o download da versão adequada do pacote py2exe, execute a instalação.

## Criando nosso primeiro executável

Vamos criar nosso primeiro programa executável a partir do script:



Note que optei por salvar o arquivo em diretório logo abaixo do C:\ (C:\aula\alo.py). Fiz esta escolha porque precisaremos utilizar o *prompt* de comando mais adiante, e para quem não possui experiência com essa ferramenta, será mais fácil de encontrar o diretório onde o arquivo foi salvo. Aconselho também a não usar nenhum caracter acentuável ou espaço em branco no nome do arquivo ou do diretório escolhido. Note ainda que usei a função *raw\_input* ao final do *script* para que o programa seja obrigado a esperar por uma entrada do usuário antes de finalizar sua execução. Sem esta linha, o programa gerado poderia funcionar muito rapidamente, sem dar ao usuário a possibilidade de vê-lo executar.

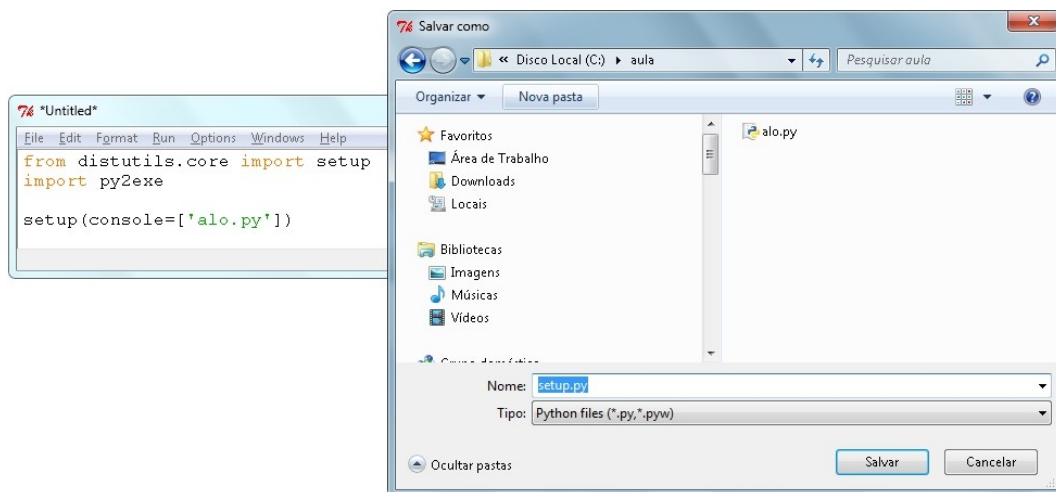
Precisamos agora escrever outro *script* para a geração do executável:

```
from distutils.core import setup
import py2exe

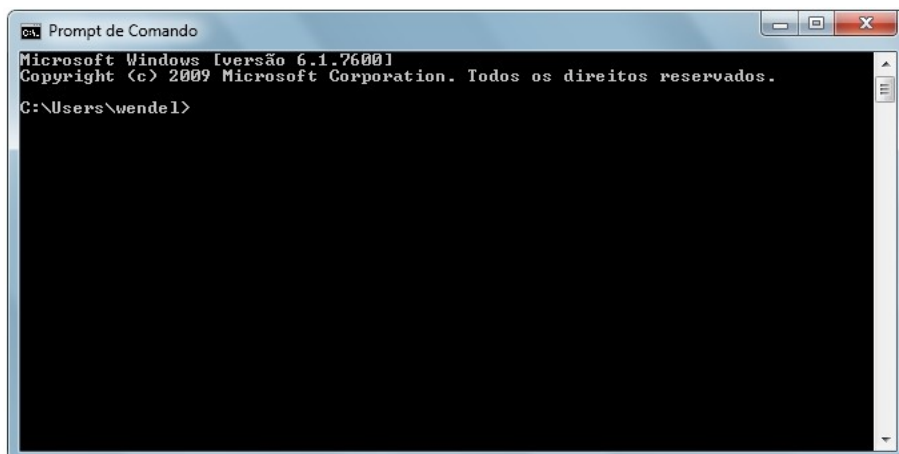
setup(console=['alo.py'])
```

Salvamos este *script* como setup.py, na mesma pasta do nosso programa alo.py

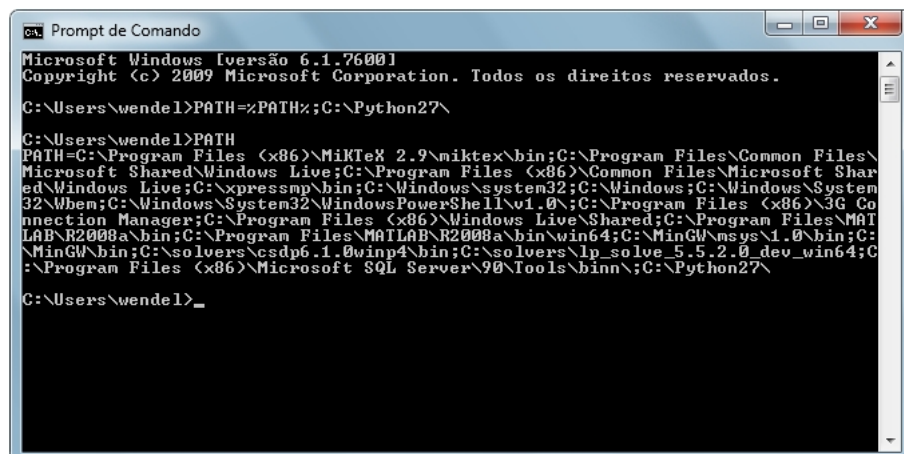
(C:\aula):



Agora temos que rodar o *script* setup.py passando-o como parâmetro para o interpretador. Para isso, usaremos o velho e bom *prompt* de comando do Windows, disponível em Iniciar >> Programas >> Acessórios >> Prompt de Comando:

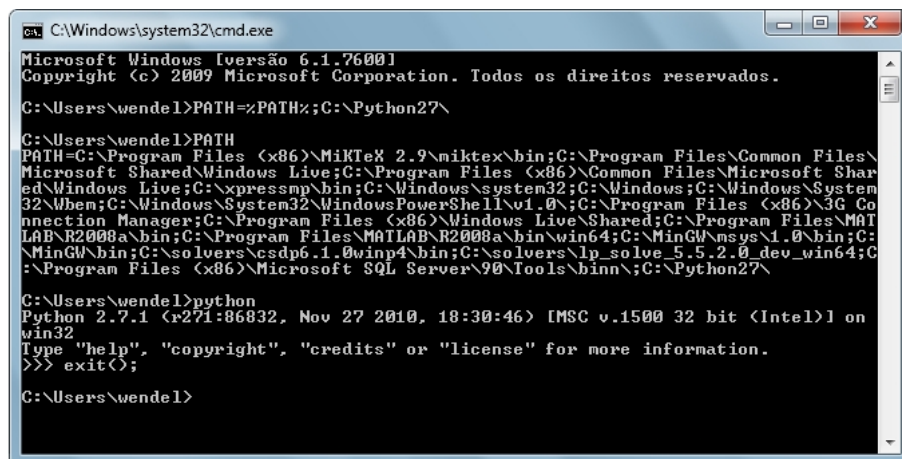


O primeiro passo é adicionar o diretório de instalação do Python à variável PATH do sistema. Esta variável PATH contém uma lista de diretórios onde o sistema busca por arquivos. Supondo que sua instalação do Python está no diretório C:\Python27, digite o comando (sem espaços em branco): "PATH=%PATH%;C:\Python27\" e pressione ENTER:



Se tudo ocorrer bem, se você solicitar o conteúdo da variável PATH (conforme feito na figura anterior), você deverá ver o diretório de instalação do Python ao final da lista. A vantagem de utilizar o *prompt* de comando para manipular o PATH é que as alterações feitas só valerão para essa seção do *prompt*. Assim, se você alterar o PATH de forma equivocada, poderá simplesmente fechar a janela do *prompt* e recomeçar o processo abrindo uma nova sem prejuízos ao sistema.

Para testar se tudo ocorreu bem, pode-se chamar o interpretador Python pelo *prompt* de comando simplesmente digitando o comando "python". Para sair do interpretador, pode-se utilizar a função `exit()`:



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versão 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Todos os direitos reservados.

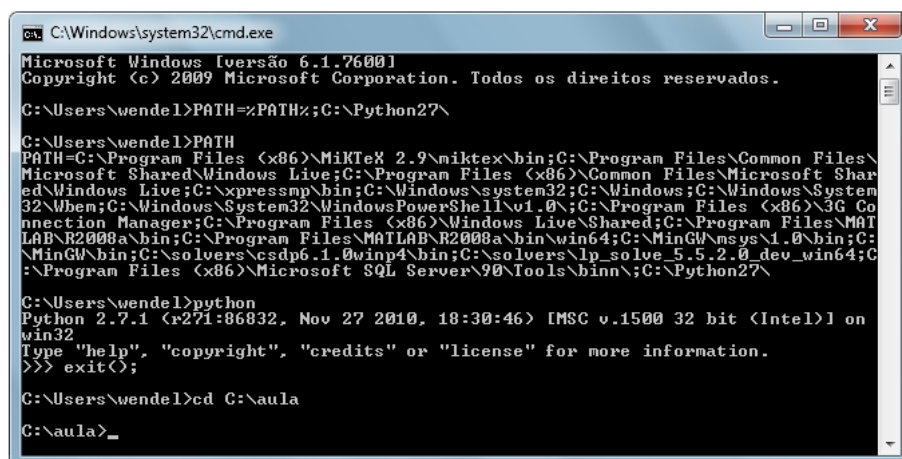
C:\Users\wendel>PATH=%PATH%;C:\Python27\

C:\Users\wendel>PATH
PATH=C:\Program Files (x86)\MikTeX 2.9\miktex\bin;C:\Program Files\Common Files\Microsoft Shared\Windows Live;C:\Program Files (x86)\Common Files\Microsoft Shared\Windows Live;C:\xpressmp\bin;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Program Files (x86)\3G Connection Manager;C:\Program Files (x86)\Windows Live\Shared;C:\Program Files\MATLAB\R2008a\bin;C:\Program Files\MATLAB\R2008a\bin\win64;C:\MinGW\msys\1.0\bin;C:\MinGW\bin;C:\solvers\csdp6.1.0\winp4\bin;C:\solvers\lp_solve_5.5.2.0_dev_win64;C:\Program Files (x86)\Microsoft SQL Server\90\Tools\bin\;C:\Python27\

C:\Users\wendel>python
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()

C:\Users\wendel>
```

Note que o diretório corrente no *prompt* é, no meu caso, C:\Users\wendel. Vamos mudar o diretório corrente para o diretório onde estão os nossos *scripts* `ola.py` e `setup.py` (C:\aula) com o comando "cd C:\aula":



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versão 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Todos os direitos reservados.

C:\Users\wendel>PATH=%PATH%;C:\Python27\

C:\Users\wendel>PATH
PATH=C:\Program Files (x86)\MikTeX 2.9\miktex\bin;C:\Program Files\Common Files\Microsoft Shared\Windows Live;C:\Program Files (x86)\Common Files\Microsoft Shared\Windows Live;C:\xpressmp\bin;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Program Files (x86)\3G Connection Manager;C:\Program Files (x86)\Windows Live\Shared;C:\Program Files\MATLAB\R2008a\bin;C:\Program Files\MATLAB\R2008a\bin\win64;C:\MinGW\msys\1.0\bin;C:\MinGW\bin;C:\solvers\csdp6.1.0\winp4\bin;C:\solvers\lp_solve_5.5.2.0_dev_win64;C:\Program Files (x86)\Microsoft SQL Server\90\Tools\bin\;C:\Python27\

C:\Users\wendel>python
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit()

C:\Users\wendel>cd C:\aula
C:\aula>
```

Note que o diretório corrente foi alterado com sucesso para C:\aula. Agora geramos o executável com o comando: "python setup.py py2exe":



```
C:\Windows\system32\cmd.exe
Microsoft Windows [versão 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. Todos os direitos reservados.

C:\Users\wendel>PATH=%PATH%;C:\Python27\

C:\Users\wendel>PATH
PATH=C:\Program Files (x86)\MikTeX 2.9\miktex\bin;C:\Program Files\Common Files\Microsoft Shared\Windows Live;C:\Program Files (x86)\Common Files\Microsoft Shared\Windows Live;C:\xpressmp\bin;C:\Windows\system32;C:\Windows;C:\Windows\System32\Wbem;C:\Windows\System32\WindowsPowerShell\v1.0\;C:\Program Files (x86)\3G Connection Manager;C:\Program Files (x86)\Windows Live\Shared;C:\Program Files\MATLAB\R2008a\bin;C:\Program Files\MATLAB\R2008a\bin\win64;C:\MinGW\msys\1.0\bin;C:\MinGW\bin;C:\solvers\csdp6.1.0\winp4\bin;C:\solvers\lp_solve_5.5.2.0_dev_win64;C:\Program Files (x86)\Microsoft SQL Server\90\Tools\bin\;C:\Python27\

C:\Users\wendel>python
Python 2.7.1 (r271:86832, Nov 27 2010, 18:30:46) [MSC v.1500 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> exit();

C:\Users\wendel>cd C:\aula

C:\aula>python setup.py py2exe_
```

Após entrar com o comando anterior, o script setup.py deve imprimir uma série de informações na tela e gerar o nosso bendito programa executável:

```
C:\Windows\system32\cmd.exe
skipping byte-compilation of C:\Python27\lib\weakref.py to weakref.pyc
byte-compiling C:\aula\build\bdist.win32\winexe\temp\_hashlib.py to _hashlib.pyc
byte-compiling C:\aula\build\bdist.win32\winexe\temp\bz2.py to bz2.pyc
byte-compiling C:\aula\build\bdist.win32\winexe\temp\select.py to select.pyc
byte-compiling C:\aula\build\bdist.win32\winexe\temp\unicodedata.py to unicodedata.pyc
*** copy extensions ***
*** copy dlls ***
copying C:\Python27\lib\site-packages\py2exe\run.exe -> C:\aula\dist\alo.exe

*** binary dependencies ***
Your executable(s) also depend on these dlls which are not included,
you may or may not need to distribute them.

Make sure you have the license if you distribute any of them, and
make sure you don't distribute files belonging to the operating system.

WS2_32.dll - C:\Windows\system32\WS2_32.dll
SHELL32.dll - C:\Windows\system32\SHELL32.dll
USER32.dll - C:\Windows\system32\USER32.dll
ADVAPI32.dll - C:\Windows\system32\ADVAPI32.dll
KERNEL32.dll - C:\Windows\system32\KERNEL32.dll

C:\aula>
```

Se tudo ocorrer bem, serão criados dois novos diretórios dentro do diretório aula: "build" e "dist". O diretório build possui arquivos utilizados no processo de construção do programa executável, os quais não nos interessam muito. O diretório dist conterá o programa executável "alo.exe". Se dermos um duplo clique sobre o mesmo, o executaremos sem o uso do interpretador Python:

```
C:\aula\dist\alo.exe
Ola Wendel!
pressione ENTER para encerrar o programa...
```

Pronto! Agora você já sabe como pode distribuir seus programas Python para os seus amiguinhos e impressionar a todos com suas habilidades em programação. É

importante mencionar que os demais arquivos na pasta dist também são importantes para o correto funcionamento do programa criado e devem ser distribuídos junto com o executável. Então, se quiser passar seu programa para terceiros, o ideal é passar todo o diretório dist gerado (pode ser em um arquivo zip, ou algo assim).

O processo aqui descrito também funciona com programas Python que utilizam interface gráfica com Tkinter. Apenas é recomendável modificar o argumento da função setup para windows ao invés de console no script setup.py:

```
from distutils.core import setup
import py2exe

setup(windows=['programaGrafico.py'])
```

Uma série de opções podem ser utilizadas para a construção do executável, incluindo a definição de um ícone para o programa. Você pode encontrar mais informações no site oficial do pacote py2exe, na web e, é claro, na ajuda do pacote (importe o pacote e chame a função help(py2exe) ).

## Resumo do processo

Eis aqui um resumo de todo o processo para a geração do nosso executável:

1. Escrever o programa Python que será usado para gerar o executável.
2. Assegurar que o pacote py2exe está corretamente instalado.
3. Escrever um *script* que guiará o processo de construção do executável (no nosso caso, foi o setup.py).
4. Abrir o *prompt* de comando e adicionar o diretório de instalação do Python ao Path do sistema.
5. Mudar o diretório corrente do *prompt* para o diretório onde estão salvos o *script* do programa e o *script* de construção do executável.
6. Chamar o interpretador Python passando como argumento o nome do *script* de construção do executável e o comando py2exe.