Antonio Sérgio Nogueira

Programando em Python®
Do Básico à WEB
Nesta edição: acesso a internet e seus serviços

2a. Edição
2012

Sumário

Antonio Sérgio Nogueira

Introdução

Este texto não tem a pretensão de ser uma bíblia da linguagem Python, mas ele aborda a linguagem em sua extensão, mostrando os aspectos mais peculiares como:

- O interpretador Python e o ambiente para edição e execução IDLE;
- A estrutura básica da linguagem e seus tipos de dados;
- A definição de bloco por indentação ao invés de marcadores de início e fim de bloco;
- A orientação a objetos;
- A interface gráfica Tkinter;
- E por fim como desenvolver um programa para Web.

Lançada por Guido van Rossum em 1991, o Python atualmente possui um modelo de desenvolvimento comunitário e aberto, gerenciado pela organização sem fins lucrativos Python Software Foundation. Python é simples de usar, e é uma verdadeira linguagem de programação que oferece tipos nativos de alto nível como flexíveis vetores, matrizes e dicionários. Por ser uma linguagem interpretada de alto nível, orientada a objetos e de tipagem dinâmica e forte. Python é muito mais fácil de usar do que as linguagens compiladas e o interpretador pode ser usado interativamente, o que torna fácil experimentar diversas características da linguagem, escrever e testar programas e funções em um desenvolvimento *bottom-up*.

O nome da linguagem não tem nada a ver com os famosos répteis e sim com o famoso show da BBC "Monty Python's Flying Circus". Mas neste exato momento se você

ainda não se convenceu com a linguagem Python, saiba que a Nasa e a Industrial Light & Magic, a companhia de efeito visual que criou Star Wars, usa Python. Agora que você já se convenceu, que tal continuar a examinar o texto com maior detalhe e testar o interpretador Python com os exemplos mostrados no texto.

O Python e sua extensa biblioteca padrão estão disponíveis na forma de código fonte ou binário para a maioria das plataformas a partir do site, http://www.python.org/, e deve ser distribuídos livremente. No mesmo sítio estão disponíveis distribuições e referências para diversos módulos, programas, ferramentas e documentação adicional contribuídos por terceiros. Acesse as documentações através de http://python.org/doc/ onde você vai encontrar: Python Library Reference, Python Reference Manual, Extending and Embedding the Python Interpreter, Python/C API Reference.

Neste livro os exemplos de programas estão dentro de quadros, e as palavras podem não estar acentuadas para evitar possíveis erros na hora de usarmos o interpretador Python.

Antonio Sérgio Nogueira

1. O interpretador Python

1.1 Sua história

No final de 1989 Guido van Rossum criou o Python no Instituto de Pesquisa nacional para Matemática e Ciência da Computação (CWI), nos Países Baixos, um sucessor da linguagem chamada de ABC. Tendo como foco principal o aumento de produtividade do programador, em 1991, Guido publicou o código (versão 0.9.0) no grupo de discussão *alt.sources*. Em 1994 formou-se o principal fórum de discussão do Python, *comp.lang.python*, isto foi o marco para o crescimento de usuários da linguagem. A versão 1.0 foi lançada em janeiro de 1994. Além das funcionalidades que já estavam presentes como classes com herança, tratamento de exceções, funções e os tipos de dados nativos, lista, dicionários e strings, sistema de módulos e assim por diante, esta nova versão incluía ferramentas para programação funcional como lambda, `map`, filter e reduce. A última versão que Guido lançou enquanto estava na CWI foi o Python 1.2. Guido continuou o trabalho, em 1995, no CNRI em Reston, USA, de onde lançou diversas versões. Na versão 1.4 a linguagem ganhou parâmetros nomeados[1] e suporte nativo a números complexos, assim como uma forma de encapsulamento (Versão 1.6 última versão lançada no CNRI)

A partir de 2000, o desenvolvimento da linguagem se mudou para a BeOpen afim de formar o time PythonLabs. O único lançamento na BeOpen foi o Python 2.0, e após o lançamento o grupo de desenvolvedores da PythonLabs agrupou-se na Digital Creations. O Python 2.0 implementou list

1 A capacidade de passar parâmetro pelo nome e não pela posição na lista de parâmetros.

comprehension, uma relevante funcionalidade de linguagens funcionais como SETL e Haskell, e essa versão 2.0 também introduziu um sistema coletor de lixo capaz de identificar e tratar ciclos de referências. O lançamento incluiu a mudança na especificação para suportar escopo aninhado, assim como outras linguagens com escopo estático. Uma grande inovação da versão 2.2 foi a unificação dos tipos Python (escritos em "C") e classes (escritas em Python) em somente uma hierarquia, com isto o modelo de objetos do Python torna-se consistentemente orientado a objeto e foi adicionado generator, inspirado em Icon. O incremento da biblioteca padrão e as escolhas sintáticas foram fortemente influenciadas por Java em alguns casos: o pacote `logging` introduzido na versão 2.3, o analisador sintático SAX, introduzido na versão 2.0 e a sintaxe de decoradores que usa `@`, adicionadas na versão 2.4. Em 1 de outubro de 2008 foi lançada a versão 2.6, já visando a transição para a versão 3.0 da linguagem. Entre outras modificações, foram incluídas bibliotecas para multiprocessamento, JSON, E/S, além de uma nova forma de formatação de cadeia de caracteres.
Atualmente a linguagem é usada em diversas áreas, como servidores de aplicação e computação gráfica. Está disponível como linguagem script em aplicações como Open Office (Python UNO Bridge) e pode ser utilizada em procedimentos armazenados no sistema gerenciador de banco de dados PostgreSQL(PL/Python). [WIKIPEDIA – 27/11/200].

Existem diversas implementações do PYTON são elas:

Cpython – esta é a versão original escrita em "C".

ActivePython - é um padrão industrial do Python, disponível para Windows, Linux, Mac OS X, Solaris, AIX e HP-UX.

Jython – esta é a versão escrita em Java e pode ser usada em implementações que usam a biblioteca de classes Java. Veja o Website do Jython.

Python for .NET – usa a implementação Cpython, mas é gerenciada por uma aplicação .NET e disponibiliza as bibliotecas .NET. Veja Python for .NET

IronPython – Diferente do Python .Net , ela é uma implementação completa de Python que gera a Linguagem Intermediária, e compila o código Python diretamente para o assembler .NET.

PyPy - Uma implementação do python escrita em Python, até o interpretador de bytecode é escrito em Python. Ela é executada usando o Cpython como um interpretador subjacente. Um dos objetivos do projeto é incentivar a experimentação com a linguagem própria, tornando-a mais fácil de modificar o interpretador (uma vez que é escrito em Python). Informação adicional está disponível na página inicial do projeto PyPy .

1.2 Licença

No que tange as licenças de uso a versão 1.6.1 do Python passou a ser compatível com a GPL, e na versão 2.1 a licença foi renomeada para Python Foundation License, em 2001. Todos os direitos de propriedade intelectual adicionados deste ponto em diante, começando com Python 2.1 e suas versões alfa e beta, são de titularidade da Python Software Foundation (PSF), uma associação sem fins lucrativos organizada sob inspiração da

Apache Software Foundation (ASF). Veja http://www.python.org/psf/ para mais informações sobre PSF.

TERMOS E CONDIÇÕES PARA ACESSAR OU DE OUTRA FORMA UTILIZAR PYTHON

==========

CONTRATO DE LICENÇA PSF

==========

1. Este CONTRATO DE LICENÇA realiza-se entre Python Software Foundation (“PSF”) e o Indivíduo ou Organização (“Licenciado”) acessando ou de outra forma utilizando o programa de computador Python 2.1.1 em forma binária ou código-fonte e sua documentação associada.
2. Nos termos e condições deste Contrato de Licença, PSF outorga ao Licenciado, por este instrumento, uma Licença não exclusiva, sem encargos patrimoniais (royalties), de abrangência mundial para reproduzir, analisar, testar, executar ou expor publicamente, preparar obras derivadas, distribuir e de outra forma utilizar Python 2.1.1, isolado ou em qualquer versão derivada, contanto que, porém, este Contrato de Licença PSF e o aviso de direitos autorais PSF, i.e., “Copyright © 2001 Python Software Foundation, All Rights Reserved” sejam incluídos em Python 2.1.1, isolado ou em qualquer versão derivada preparada pelo Licenciado.
3. Caso o Licenciado prepare uma obra derivada baseada em Python 2.1.1 ou que o incorpore por

inteiro ou qualquer trecho seu, e deseje tornar esta obra derivada disponível a outrem como aqui previsto, então o Licenciado concorda em incluir em tal obra um breve sumário das mudanças feitas sobre Python 2.1.1.

4. PSF torna Python 2.1.1 disponível ao Licenciado "COMO ESTÁ". PSF NÃO OFERECE QUAISQUER GARANTIAS OU DECLARAÇÕES, EXPRESSAS OU TÁCITAS. COMO EXEMPLO, MAS NÃO LIMITAÇÃO, PSF NÃO OFERECE E SE ISENTA DE QUAISQUER GARANTIAS OU DECLARAÇÕES DE COMERCIABILIDADE OU ADEQUAÇÃO A FINALIDADES ESPECÍFICAS, OU DE QUE O USO DE PYTHON 2.1.1 NÃO VIOLARÁ QUAISQUER DIREITOS DE TERCEIROS.

5. PSF NÃO SERÁ RESPONSÁVEL PERANTE O LICENCIADO OU QUAISQUER OUTROS USUÁRIOS DE PYTHON 2.1.1 POR PERDAS E DANOS, SEJAM INCIDENTAIS, ESPECIAIS OU CONSEQUENTES, COMO RESULTADO DE MODIFICAÇÃO, DISTRIBUIÇÃO, OU OUTRA FORMA DE UTILIZAÇÃO DE PYTHON 2.1.1, OU QUALQUER DE SUAS OBRAS DERIVADAS, MESMO QUE HOUVESSE SIDO AVISADA DESTA POSSIBILIDADE.

6. Este Contrato de Licença será automaticamente rescindido em caso de violação material de seus termos e condições.

7. Nada neste Contrato de Licença pode ser interpretado de forma a criar qualquer relação de

agência, parceria ou joint-venture entre PSF e o Licenciado. Este Contrato de Licença não outorga permissão para usar marcas ou nomes comerciais de PSF como conjunto distintivo para endossar ou promover produtos ou serviços do Licenciado ou de qualquer terceiro.

8. Ao copiar, instalar ou de outra forma utilizar Python 2.1.1, o Licenciado obriga-se aos termos e condições deste Contrato de Licença.

==========

1.3 Uso do Interpretador Python

1.3.1 Qual Python usar?

O Windows não vem com Python. Mas não se preocupe! Há diversas formas fáceis de entrar no mundo Python usando Windows. Como você pode ver, Python roda em muitos sistemas operacionais. A lista completa inclui Windows, o Mac OS, o Mac OS X e todos os sistemas gratuitos compatível com UNIX como o próprio Linux. Há também versões que rodam no Sun Solaris, AS/400, Amiga, OS/2, BeOS e muitas outras plataformas que você provavelmente nunca ouviu falar.

1.3.2 Python em Windows

No Windows, você tem diversas escolhas para instalar o Python. A ActiveState tem um instalador Python para Windows que inclui a versão completa do Python, um IDE com um editor de código e extensões Windows para o Python que permitem acesso a serviços específicos do Windows, suas APIs e o

registro. O ActivePython pode ser baixado gratuitamente, mas não é open source. Se você realmente precisa da última versão do Python pule para a opção 2. A segunda opção é usar o instalador Python "oficial", distribuído pelos próprios desenvolvedores do Python. Esse instalador pode ser baixado gratuitamente, tem código fonte aberto e está sempre atualizado.

1.3.2.1 Opção 1: Instalando o ActivePython

1. Baixe o ActivePython em http://www.activestate.com/Products/ActivePython/
2. Se você usa Windows 95, Windows 98 ou Windows ME, deve instalar o Windows Installer 2.0 antes de continuar.
3. Dê um clique duplo no botão com o arquivo ActivePython 2.7.2 for Windows(x86).
4. Siga as instruções na tela.
5. Se seu espaço em disco for limitado, é possível fazer uma instalação personalizada ("custom") e deixar de instalar a documentação, mas isso não é recomendado.
6. Após o término da instalação, feche o instalador e abra Iniciar->Programas->ActiveState ActivePython 2.7(32-bit)->IDLE(Python GUI).

Example IDE ActivePython

ActivePython 2.7.2.5 (ActiveState Software Inc.) based on
Python 2.7.2 (default, Jun 24 2011, 12:21:10) [MSC v.1500 32 bit (Intel)] on win32
Type "copyright", "credits" or "license()" for more information.
>>>

1.3.2.2 Opção 2: Instalando o Python do Python.org

1. Baixe o instalador Windows do Python em
http://www.python.org/download/releases/2.7.2/
2. Execute o arquivo Python-2.7.3 Windows Installer
3. Siga as instruções na tela.
4. Após o término da instalação, feche o instalador e abra Iniciar->Programas->Python 2.7->IDLE
(Python GUI).

Figura 1: IDLE (Python GUI)

2. Python: Introdução, Tipos e Comandos Básicos

2.1 Interpretador

Python é uma linguagem interpretada, o que significa que o código não precisa ser compilado para que seja executado. Assim, o interpretador lê e executa o código diretamente. Linguagens interpretadas, normalmente, funcionam através de 'Compilação Just- In-Time' ou 'Interpretação pura ou em Bytecode'. Você pode criar seu arquivo Python e salvá-lo com a extensão ".py" ou ainda pode executar no modo shell, ou seja, você digita o código diretamente no interpretador. Python é multiplataforma, roda em Windows, Linux, Unix, Macintosh, etc. Pode ser utilizado um ambiente para edição e execução como o IDLE, que é nativo da linguagem e para acessá-lo em Windows basta ir em INICIAR>PROGRAMAS>PYTHON 2.7>IDLE. Em computadores com Windows, Python é instalado geralmente em 'C:\Python27', apesar de você poder mudar isso enquanto está executando o instalador. Para adicionar esse diretório ao *path*, você pode digitar o seguinte comando no DOS: set path=%path%;C:\python27

2.2 Palavras Reservadas

São palavras que não podem ser usadas para dar nomes aos objetos. São: **and, assert, break, class, continue, del, def, elif, else, except, exec, finally, for, from, global, if, import, in, is, lambda, not, or, pass, print, raise, return, try, while.**

2.3 Digitando comandos no IDLE

Podemos usar o IDLE como uma calculadora,

visualizando resultados e usando os resultados em cálculos futuros.

```
>>> a=1
>>> b=2
>>> a+b
>>> a/b
>>> a*b
```

No modo interativo, a última expressão a ser impressa é atribuída a variável[2] “ _ ” . Isso significa que ao utilizar Python como uma calculadora, é muitas vezes mais fácil prosseguir com os cálculos da seguinte forma:

```
>>> taxa = 12.5 / 100
>>> preco = 100.50
>>> preco* taxa
12.5625
>>> preco + _
113.0625
>>> round(_, 2)
113.06
```

Essa variável especial deve ser tratada somente para leitura pelo usuário. Nunca lhe atribua explicitamente um valor, do contrário estaria criando uma outra variável (homônima) independente, que mascararia o comportamento mágico da variável especial.

2 Local de memória usado para guardar valores.

2.4 Tipos de Dados

São categorias de valores que são processados de forma semelhante. Por exemplo, números inteiros são processados de forma diferente dos números de ponto flutuante(decimais) e dos números complexos.
Tipos primitivos: são aqueles já embutidos no núcleo da linguagem
Simples: números (int, long, float, complex) e cadeias de caracteres (strings)

Int: números inteiros de precisão fixa
1 , 2 , 15 , 19
Long: números inteiros de precisão arbitrária
1L , 10000L , -9999999L

Floats: números racionais de precisão variável.
1.0 , 10.5 , -19000.00005 , 15e-5

Complex: números complexos.
1+1j , 20j , 1000+100J

Compostos: listas, dicionários, tuplas e conjuntos.

Tipos definidos pelo usuário: são correspondentes a classes (orientação objeto).

2.5 Número em ponto flutuante e complexo

Há total suporte para ponto flutuante; operadores com operandos de diferentes tipos convertem o inteiro para ponto flutuante:

```
>>> 3 * 3.75 / 1.5
7.5
>>> 7.0 / 2
3.5
```

Números complexos também são suportados; números imaginários são escritos com o sufixo ‘j’ ou ‘J’. Números complexos com parte real não nula são escritos como ‘(*real*+*imag*j)’, ou podem ser criados pela chamada de função ‘complex(*real*, *imag*)’.

```
>>> 1j * 1J
(-1+0j)
>>> 1j * complex(0,1)
(-1+0j)
>>> 3+1j*3
(3+3j)
>>> (3+1j)*3
(9+3j)
>>> (1+2j)/(1+1j)
(1.5+0.5j)
```

Números complexos são sempre representados por dois números em ponto flutuante, a parte real e a parte imaginária. Para extrair as partes de um número *z*, utilize z.real e z.imag.

```
>>> a=1.5+0.5j
>>> a.real
1.5
>>> a.imag
0.5
```

As funções de conversão para ponto flutuante e inteiro, float(), int() e long() não funcionam para números complexos, pois não existe maneira correta de converter um número complexo para um número real. Utilize abs(z) para obter sua magnitude (como ponto flutuante) ou z.real para obter sua parte real.

```
>>> a=3.0+4.0j
>>> float(a)
Traceback (most recent call last):
File "<stdin>", line 1, in ?
TypeError: can't convert complex to float; use abs(z)
>>> a.real
3.0
>>> a.imag
4.0
>>> abs(a)
5.0
```

2.6 Strings

Além de números, Python também pode manipular strings[3]. O IDLE não está preparado para trabalhar com caracteres acentuados. Durante a execução de um programa em Python, use este comando na primeira linha para acentuar os caracteres: **# -*- coding: cp1252 -*-**

3 Strings são sequências de caracteres entre aspas ou apóstrofo, modificáveis apenas com atribuição de novos valores.

```
>>> ’mais menos’
’mais menos’

>>> "vida's"
"vida's"
```

```
>>> '"OLA", BOM DIA.'
'"OLA", BOM DIA.'

>>> "\"OLA,\" BOM DIA."
'"OLA," BOM DIA.'
```

Strings que contém mais de uma linha podem ser construídas de diversas maneiras, uma delas é terminadores de linha embutidos na string (“\”).

```
# -*- coding: cp1252 -*-
#Programa que imprime oi
oi = "Esta é uma string longa contendo\n\
diversas linhas de texto assim como você faria em C.\n\
 Observe que os espaços em branco no inicio da linha são \
significativos."
print oi

Resultado da execução:

Esta é uma string longa contendo
diversas linhas de texto assim como você faria em C.
 Observe que os espaços em branco no inicio da linha são significativos.
```

Observe que terminadores de linha ainda precisam ser embutidos na string (“\n”); a quebra de linha após a última barra de escape é ignorada.

No entanto, se a tornarmos uma string “crua” (*raw*), as sequências de \n não são convertidas para quebras de linha. Tanto a barra invertida (“\”) quanto a quebra de linha (“\n”) no código-fonte são incluídos na string como dados. Portanto, o exemplo:

```
>>>oi = r"Esta eh uma string longa contendo\n\
diversas linhas de texto assim como voce faria em C.\n\
 Observe que os espaços em branco no inicio da linha são \
significativos."
>>>print oi

Resultado:
Esta eh uma string longa contendo\n\
diversas linhas de texto assim como voce faria em C.\n\
   Observe que os espaços em branco no inicio da linha são \
significativos.
```

As strings também podem ser delimitadas por pares de aspas ou apóstrofos tríplices: **”””**ou **’’’**. Neste caso não é necessário embutir terminadores de linha, pois o texto da string será tratado fielmente como o original.

```
>>> print """ ola
tudo bem e ai como
vai."""
```

```
Resultado:
 ola
tudo bem e ai como
vai.
```

O interpretador imprime o resultado de operações sobre strings da mesma forma que as strings são formatadas na digitação: dentro de aspas, e com caracteres especiais embutidos em *escape sequences*[4], para mostrar seu valor com precisão a string será delimitada por aspas duplas, se ela contém um único caractere de aspas simples e nenhum de aspas duplas, caso contrário a string será delimitada por aspas simples. Strings podem ser concatenadas (coladas) com o operador +, e repetidas com *:

```
>>> palavra='palavra '+'A'
>>> palavra
'palavra A'

>>> '<'+palavra * 5 + '>'
'<palavra Apalavra Apalavra Apalavra Apalavra A>'

>>>print "oi" 'vida'
oi vida
>>> print 'oi' 'vida' # Duas strings literais justapostas são automaticamente concatenadas.
oivida
>>>
```

Strings podem ser indexadas; como em C, o primeiro

4 *Caractere especial como \n – nova linha*

índice da string é o 0. Não existe um tipo separado para caracteres; um caractere é simplesmente uma string unitária. Assim como na linguagem Icon, substrings[5] podem ser especificadas através da notação *slice*[6]: dois índices separados por dois pontos.

```
>>> palavra[4]
'v'
>>> palavra[0:2]
'pa'
>>> palavra[2:4]
'la'
```

Índices de fatias seguem uma padronização útil; a omissão do primeiro índice equivale a zero, a omissão do segundo índice equivale ao tamanho da string que está sendo fatiada.

```
>>> palavra[:2] # os dois primeiros caracteres
'pa'

>>> palavra[2:]  # todos menos os 2 primeiros
'lavra A'

>>> palavra[-1] # último caractere
'A'

>>> palavra[:-2] # todos menos os 2 últimos
'palavra'
```

5 Uma pedaço da string.

6 Tradução: Fatiar,ou cortar em pedaços. Notação slice: palavra[2:4]