

# *Python* na Prática

Um tutorial sobre a linguagem Python, conceitos e sintaxe com uma abordagem prática e direta.

Celso Provedelo <[cprov@canonical.com](mailto:cprov@canonical.com)>

Canonical Ltd.



# O que é Python

- ✓ Linguagem interpretada
- ✓ Claridade e Simplicidade
- ✓ Blocos controlados por Indentação
- ✓ Tipagem dinâmica
- ✓ Tipos avançados
- ✓ Orientação a objetos



# Porque Python

- ✓ Conceitos simples
- ✓ Sintaxe simples, clara e expressiva
- ✓ Poderosos tipos pré-definidos
- ✓ Interpretador interativo
- ✓ Várias bibliotecas
- ✓ Extensões em C e C++
- ✓ Multi-plataforma
- ✓ Software Livre



# Python básico: Interpretador

- Interpretador Interativo
  - ✓  $2 + 2$
  - ✓ *print*
  - ✓ Escape: *CTRL-D*
- Criando e executando código
  - ✓ *Unix-like: #!/usr/bin/env python*
  - ✓ *Win32: HelloWorld.py*

# Python básico: Tipos, variáveis e valores

- ✓ Numéricos:  $a = 2$ 
  - *int, float, bool, complex*
- ✓ Listas:  $L = [1, 2, 3, 5, 8]$
- ✓ Strings:  $S = \text{"Hello World"}$
- ✓ Tuplas:  $T = (\text{"Romeo"}, \text{"Juliet"})$
- ✓ Dicionários:  
 $D = \{\text{"name": "Karl"}, \text{"age": 28}\}$
- ✓ Set:  $\text{set}([1, 2, 3])$
- ✓ *type(var)*

# Python básico: Operadores

- ✓ Aritiméticos: +, -, /, \*, %, \*\*
- ✓ Bit-wise: &, |, <<, >>, ~
- ✓ Atribuição: =, +=, -=, /=, \*=
- ✓ Condicionais: ==, !=, <, >, <=, >=, in
- ✓ Lógicos: and, or, not, is
- ✓ Combinando operadores
- ✓ Substituição em strings

# Python básico: Estruturas de Controle

- ✓ Condicional: `if a == b`
- ✓ Laço iterativo:
  - ✓ `for l in L, for s in S, for k in D.keys()`
  - ✓ `break & continue`
  - ✓ `else`
- ✓ Laço condicional:
  - ✓ `while i < 5`
  - ✓ `break`
  - ✓ `else`

# Python básico: Funções

- ✓ Sintaxe: `def foo(args)`
  - ✓ `def desculpa(motivo):`
- ✓ Valor padrão
  - ✓ `def desculpa(motivo="doente"):`
- ✓ Manipulando argumentos: `*args, **kw`
  - ✓ `def desculpa(*args):`
  - ✓ `def desculpa(**kw):`
- ✓ Escopo Local & Global

# Python básico: Funções padrão - I

- ✓ `range(a, b)`
- ✓ `len(L)`
- ✓ `round(f, p)`
- ✓ `chr(i) / ord(c) / unichr(i)`
- ✓ `min(a, b) / max(a, b) / abs(n)`
- ✓ `hex(n) / oct(n) / float(n) / int(f)`
- ✓ `list(args) / str(args)`

# Python básico: Funções padrão - II

- ✓ `open()`
  - ✓ `read()`
  - ✓ `readline()`
  - ✓ `readlines()`
  - ✓ `write()`
  - ✓ `seek()`
  - ✓ `close()`
- ✓ `raw_input()`

# Python básico:Exceções

- ✓ Conceitos
  - ✓ O que é uma Exceção
  - ✓ Como uma exceção é emitida
  - ✓ tracebacks
- ✓ Manipulando exceções:
  - ✓ try/except
  - ✓ else

# Python básico: *Import* e Docstring

- ✓ *import os*
- ✓ PYTHONPATH
- ✓ *dir()* & *reload()*
  - ✓ *dir(os)*
- ✓ *help()* & *docstring*
  - ✓ *help(os)*
- ✓ Ferramentas de documentação: *epydoc*

# Orientação a Objetos (OO)

- ✓ Conceitos
  - ✓ OO vs. programação procedural
  - ✓ Domínio / Problema
- ✓ Objetos, Classes e Instâncias
- ✓ Herança
  - ✓ Herança Múltipla e `getattr()`
  - ✓ atributo de classe ou instância
- ✓ Introspecção, Reflexão e metadados

# Módulos Importantes

- *sys*
- *time*
- *os*
- *string*
- *math*
- *optparser*
- *urllib*
- PIL
- OpenGL
- NumPy
- pyGTK
- DB-API
- Zope
- Twisted

# Obrigado !

Agradecimentos:

Christian Reis,

Prof. Cláudio Kirner

Canonical Ltd

