Python a Robot Framework

Den první

ENGETO s.r.o.

Prezentované materiály jsou dostupné na adrese
https://github.com/ChaoticRoman/python-a-robot

Cíle kurzu

- Transfer znalostí a zkušeností z profesionální praxe Python vývojáře:
  - Porozumění Pythonu
  - Proces vývoje: specifikace, design, implementace, dokumentace, testování, distribuce
  - Testování: teorie, unittesting, mocking, acceptance testing, test and behavior driven development, Robot Framework
  - Doporučené postupy
  - Užitečné knihovny

Předpoklady

- Notebook při ruce
  - Hands-on výuka je preferována
- Znalost základů Pythonu
  - Toto není kurz pro úplné začátečníky
- Nainstalovaný Python 3 a git
  - Python 2.x přestane být oficiálně podporován v roce 2020
  - Git představuje základní nástroj současného vývojaře
- Motivace stát se profesionálním Python vývojářem
  - Nutný předpoklad :-)

Osnova kurzu

1. den: Filosofie, instalace a základy Pythonu, Python 2 a 3, standardní knihovna, objektově orientované programování, malé projekty s knihovnami numpy, matplotlib a Tkinter, distribuce aplikací pomocí setuptools a distutils

2. den: Středně velké projekty, organizace kódu, dokumentace, stanovení požadavků, teorie softwarového testování, unittesting, doporučené praktiky

3. den: Větší projekty, softwarové testování: integrační testování, systémové testování, akceptační testování, TDD, BDD, Robot Framework: základy

4. den: Kooperace pomocí gitu, logování, standardy, společný projekt,
Robot Framework: pokročilé funkce a doporučené praktiky

5. den: Testování a mocking, Robot Framework: uživatelská rozšíření, společný projekt

Den první

- Filosofie Pythonu
- Python pro Linux, Windows, Mac, mobile a embedded
- Základy Pythonu, Python 2 a 3, standardní knihovna
- Objektově orientované programování
- numpy, matplotlib, Tkinter, setuptools a distutils
- Organizace kódu

Instalace Pythonu, gitu a stažení příkladů

- Python 3
- pydoc3
- pip3
- Git
- Fork https://github.com/ChaoticRoman/python-a-robot
- git clone https://github.com/<username>/python-a-robot

Filosofie Pythonu

```
>>> import this
```

Python pro Linux, Windows, Mac, mobile a embedded

Situace na různých platformách:

Python pro Linux, Windows, Mac, mobile a embedded

Situace na různých platformách:

- Linux:
  - Python 2 zabudovaný ve většině distribucí
  - Python 3 většinou dostupný jako dodatečný balíček
  - Knihovny dostupné často přes nativní balíčkovací systém
  - pip a pip3 pro nezabalíčkované knihovny

Python pro Linux, Windows, Mac, mobile a embedded

Situace na různých platformách:

- Windows:
  - Python 2 i 3 mají oficiální instalátor
  - Anaconda distribuce
  - Python 2 i 3 v cygwinu
  - py2exe

Python pro Linux, Windows, Mac, mobile a embedded

Situace na různých platformách:

- Mac OS X:
  - Python 2 zabudovaný od Mac OS X 10.8
  - Oficiální instalátor dostupný pro Python 3
  - pip a pip3 pro knihovny třetích stran
  - py2app

Python pro Linux, Windows, Mac, mobile a embedded

Situace na různých platformách:

- mobile
  - Situace tradičně špatná
  - Malá dostupnost knihoven třetích stran
  - kivy se zdá být nejvhodnější dnes
    - Android APK i iOS app
    - ARM Android ano, x86_64 Android ne
    - Celý Python je zabalen v aplikaci

Python pro Linux, Windows, Mac, mobile a embedded

Situace na různých platformách:

- Embedded zařízení:
  - Embedded linux zařízení plně podporují Python
    - Python je výborná volba pro Raspberry Pi
  - Micropython pro pyboard, BBC microbit, ESP 8266, ESP 32 a další…
    - Vhodné pro výuku a jednoduché aplikace
    - Pro větší aplikace nevhodné, stále je zde králem C/C++

Základy Pythonu

- Základní řízení toku kontroly programu

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce
- Uživatelské funkce, argumenty funkcí, návratové hodnoty

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce
- Uživatelské funkce, argumenty funkcí, návratové hodnoty
- Kontext management

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce
- Uživatelské funkce, argumenty funkcí, návratové hodnoty
- Kontext management
- List comprehension

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce
- Uživatelské funkce, argumenty funkcí, návratové hodnoty
- Kontext management
- List comprehension
- Generátory

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce
- Uživatelské funkce, argumenty funkcí, návratové hodnoty
- Kontext management
- List comprehension
- Generátory
- Vyjímky a jejich zpracování

Základy Pythonu

- Základní řízení toku kontroly programu
- Datové typy
- Built-in funkce
- Uživatelské funkce, argumenty funkcí, návratové hodnoty
- Kontext management
- List comprehension
- Generátory
- Vyjímky a jejich zpracování
- Magické proměnné

Python 2 vs. Python 3

- print

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str
- Celočíselné a float-casting dělení: // vs. /

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str
- Celočíselné a float-casting dělení: // vs. /
- unittesting

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str
- Celočíselné a float-casting dělení: // vs. /
- unittesting
- raw_input vs. input

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str
- Celočíselné a float-casting dělení: // vs. /
- unittesting
- raw_input vs. input
- list comprehension leaks

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str
- Celočíselné a float-casting dělení: // vs. /
- unittesting
- raw_input vs. input
- list comprehension leaks
- Bankéřské zaokrouhlování

Python 2 vs. Python 3

- print
- str, unicode vs. bytes, str
- Celočíselné a float-casting dělení: // vs. /
- unittesting
- raw_input vs. input
- list comprehension leaks
- Bankéřské zaokrouhlování
- xrange vs. range

Standardní knihovna

Tour de standard library:

https://docs.python.org/3/library/index.html

Objektově orientované programování

- Syntax

Objektově orientované programování

- Syntax
- Proměnné třídy vs. proměnné instance

Objektově orientované programování

- Syntax
- Proměnné třídy vs. proměnné instance
- Metody třídy vs. metody instance

Objektově orientované programování

- Syntax
- Proměnné třídy vs. proměnné instance
- Metody třídy vs. metody instance
- Magické metody: __init__, __call__, __str__, __repr__

Objektově orientované programování

- Syntax
- Proměnné třídy vs. proměnné instance
- Metody třídy vs. metody instance
- Magické metody: __init__, __call__, __str__, __repr__
- Inheritance (dědičnost)

numpy

- Rychlé datové typy pro numerické operace
- ndarray vs. matrix
- Přehled funkcionality: https://docs.scipy.org/doc/numpy/reference/routines.html
- Scipy: https://www.scipy.org/

matplotlib

- Bohaté možnosti vizualizace
- MATLAB-like interface: matplotlib.pyplot
- Galerie: https://matplotlib.org/gallery.html

Hands-on: lineární regrese pomocí numpy a matplotlib

Tkinter

- Defakto standardní GUI knihovna pro Python
- Reference:
  - https://docs.python.org/3/library/tkinter.html
  - http://effbot.org/tkinterbook/
  - https://www.python-course.eu/tkinter_layout_management.php

Hands-on: konverter z palců do centimetrů

setuptools a distutils

- Tvorba instalátorů pro uživatelské aplikace
- setuptools nadstavba distutils
- Reference: http://python-packaging.readthedocs.io/en/latest/minimal.html

Hands-on: zabalení aplikace

Organizace kódu v malých projektech

- Top-down řazení kódu

Top-down řazení kódu

```
import argparse, configparser

def main():
    args, config_values = '', '...'  # argparse and config parse, no logic
    well_named_top_level_function(args, config_values)

def well_named_top_level_function(args, config_values):
    well_named_component_abc()

def well_named_component_abc():
    return 123

if __name__ == "__main__":
    main()
```

Organizace kódu v malých projektech

- Top-down řazení kódu

Organizace kódu v malých projektech

- Top-down řazení kódu
- KISS: Keep it stupid simple

Organizace kódu v malých projektech

- Top-down řazení kódu
- KISS: Keep it stupid simple
- YAGNI: You aren’t gonna need it

Organizace kódu v malých projektech

- Top-down řazení kódu
- KISS: Keep it stupid simple
- YAGNI: You aren’t gonna need it
- DRY: Don’t repeat yourself

Organizace kódu v malých projektech

- Top-down řazení kódu
- KISS: Keep it stupid simple
- YAGNI: You aren’t gonna need it
- DRY: Don’t repeat yourself
- PEP 8 a statická analýza: flake8

Organizace kódu v malých projektech

- Top-down řazení kódu
- KISS: Keep it stupid simple
- YAGNI: You aren’t gonna need it
- DRY: Don’t repeat yourself
- PEP 8 a statická analýza: flake8
- Vyhýbejte se příliš chytrým řádkům

Organizace kódu v malých projektech

- Top-down řazení kódu
- KISS: Keep it stupid simple
- YAGNI: You aren’t gonna need it
- DRY: Don’t repeat yourself
- PEP 8 a statická analýza: flake8
- Vyhýbejte se příliš chytrým řádkům
- O pojmenování věcí

Hands-on projekt: Vizualizace spolehlivosti

Zadání a příklad vstupních dat na githubu, složka visualization

Den první: rekapitulace

- Filosofie Pythonu
- Python pro Linux, Windows, Mac, mobile a embedded
- Základy Pythonu, Python 2 a 3, standardní knihovna
- Objektově orientované programování
- numpy, matplotlib, Tkinter, setuptools a distutils
- Organizace kódu