A.L.S.E.
http://www.alse-fr.com

The Python Language & Scientific Computing

Formation Pratique de 3 jours

Présentation

Nombreux sont nos clients (dans la Recherche puis dans l'Industrie) qui ont déjà franchi le pas et délaissé des outils propriétaires et coûteux au profit de Python et de ses librairies d'extension, notamment dans le domaine de la **modélisation et du traitement de signal**. C'est pourquoi nous avons construit ce training, avec une partie venant de notre partenaire Doulos pour le langage lui-même, et une journée dédiée aux extensions scientifiques que nous avons construite pour répondre à ces besoins.

The Python Language & Scientific Computing est donc un cours modulaire de **trois jours** qui enseigne toutes les bases indispensables pour maîtriser le **langage Python** (deux premiers jours) et les **extensions « scientifiques »** (NumPy, SciPy, MatPlotLib...). Cette formation n'exige absolument *pas que vous soyez un électronicien !* (surtout si vous vous satisfaites des deux premiers jours).

En effet, **Python** est un langage de programmation (OOP : orienté objet) moderne, très puissant et élégant, gratuit, multi-plateformes (Windows / Linux / Mac), désormais enseigné en priorité et très largement utilisé dans de nombreux domaines. Une partie de son succès s'explique par ses très nombreuses extensions de toutes natures, et en particulier dans le domaine scientifique.

Une connaissance *générale* du langage ouvre son utilisation à de nombreuses applications, dont le scripting et le pilotage d'applications tierces (qui reste encore fortement l'apanage du Tcl/Tk), ou encore le test logiciel. Mais ce ne sont aujourd'hui que des utilisations marginales.

La connaissance des principales ***extensions*** *(librairies)* ***scientifiques*** et leur application au *traitement numérique de signal* permet de faciliter la transition depuis des outils propriétaires comme MATLAB®.

Ce cours s'inscrit dans le cadre d'une **action d'acquisition** ou de **perfectionnement des connaissances.**

Les **exercices pratiques**, qui occupent environ **50 % du temps de la formation**, sont des éléments clés du succès de ces formations, ils sont indispensables pour un réel apprentissage. Ils **assurent** également **un contrôle continu de l'acquisition des compétences.**

Les **Instructeurs ALSE** sont aussi et surtout des Experts en Conception et Vérification qui utilisent à journée entière les langages qu'ils enseignent pour concevoir et vérifier des systèmes complexes, et Python fait maintenant partie de leurs outils.
Ils savent partager leur savoir-faire avec passion et sont particulièrement appréciés des participants.

Pré-requis

Aucune connaissance préalable du langage Python n'est demandée.

Par contre, une expérience préalable avec au moins un langage de programmation (C, C++, Java, Tcl/Tk, VHDL, Verilog, SystemVerilog etc...) est très fortement recommandée.

Objectifs pédagogiques

- Maîtriser le Langage Python.
- Savoir utiliser les extensions scientifiques.
- Comprendre comment transiter d'outils propriétaires vers Python notamment pour les applications de Traitement Numérique de Signal (DSP).

Qu'apprendrez-vous ?

---- *Essential Python*

- Les principes du langage Python (syntaxe, sémantique).
- Les particularités de Python
- Python Orienté Objets (classes, héritage...)
- Les Environnements de Développement (IDE)
- La Librairies Standard et ses modules les plus utilisés
- Les Expressions Régulières
- Python comme outil de Scripting (pilotage d'applications)
- Python comme outil de test pour l'embarqué

---- *Python for Scientific Computing*

- Les Librairies scientifiques : NumPy, SciPy, MatPlotLib …
- Les environnements et distributions dédiés à l'usage scientifique
- Utilisation en Traitement Numérique de Signal
- Ipython et Jupyter
- Transiter de MATLAB® , Scilab, Octave (…) vers Python

Supports de cours

Nos manuels sont réputés pour être détaillés, précis et faciles d'utilisation.

Leur style, leur contenu et leur exhaustivité sont uniques dans le monde de la formation.
Ils sont souvent utilisés ensuite comme référence.

Sont compris dans la formation :

- Le Classeur du cours théorique, avec un Index.
  Il constitue un **Manuel de Référence** concis.
- Le **Cahier des Exercices** pratiques, qui permet de mettre en œuvre les connaissances théoriques.
- Les **fichiers** des exercices et des solutions.

Voyez pages suivantes le programme détaillé de cette formation.

A.L.S.E.
http://www.alse-fr.com

Formation

The Python Language & Scientific Computing

Programme*

Partie I : Essential Python (jours 1 & 2)

Introduction

- What is Python? • The Python World
- The Python Shell
- The Zen of Python
- Running Python Programs From a File • IDLE

Les Bases du Langage

- Numbers • Strings
- Type conversions
- String Index • String Slice • String Methods
- Simple formatting

Instructions de contrôle

- Commentaires • If statements
- Comparison & boolean operators •
- For statements • Break • Continue • While statement
- Assert
- Functions Functions • Global Variables • Nonlocal Variables
- Lines and Continuation • IDLE

Listes, Tuples et Dictionnaires

- Lists • List Append, Insert, Pop
- Loops and Lists • Sorting Lists • List Comparison
- Tuples
- Dictionaries
- Sets

Formatting

- Accessing Arguments • Field Width, Justification, Padding
- Number Base, Comma, Sign
- Floating Point

Files and Exceptions

- Reading Standard Input
- Writing to a File • Reading from a File
- Variations • readline
- Exceptions • Context Manager

Classes

- Classes and Methods
- Constructors
- Bundling Data Attributes
- Class Variables and Instance Variables
- Class vs Object vs Function vs Method
- Global, Local, Class, and Instance Variables
- Documentation Strings

Inheritance

- Inheritance
- Virtual Method Calls
- Multiple Inheritance
- Testing Class Relationships
- Tying Variables to a Class
- Duck Typing

Copying Objects

- Copying Class Objects
- Copying Lists
- Shallow and Deep Copies

Iterators and Generators

- Sequence versus Iterator
- Iterables and Iterators
- Generators
- List Comprehensions
- Generator Expressions
- lambda
- map
- filter, enumerate
- zip
- join
- Dictionary Comprehensions

Exploring Functions

- Default and Keyword Arguments
- Argument Lists
- None
- Functions are Objects, def is a Statement
- Decorators
- Parameterized Decorators
- Closures

Python 2.x vs Python 3.x

- Syntax
- Type Conversion
- Strings and Bytes
- Iterables
- Arithmetic

Modules

- import
- from ... import
- __name__
- Packages

The Standard Library

- math
- random
- statistics
- datetime
- time
- timeit
- os
- shutil
- glob
- sys
- subprocess

Regular Expressions

- match
- search
- findall
- Filter the Output from Another Program
- sub
- Basic Regular Expression Syntax

Test-Driven Development

- What is TDD?
- The TDD Process
- The Four-Phase Test Pattern
- Fakes and Test Doubles

Unit Testing

- A Simple Unit Test
- From the Command Line
- Test Discovery
- Test Fixture
- Assert Methods
- Specific Error Messages
- Adding a Message Argument
- Testing Exceptions
- Subtests
- Skipping Tests
- Expected Failures
- Test Suites
- Test Suites within Suites
- Redirecting or Inspecting Results

Partie II : Scientific computing & Signal Processing (jour 3)

NB : ce contenu est nouveau et préliminaire : il est susceptible d'évoluer.

Introduction

- Python's success in Scientific and Engineering environments
- Working environments for Scientific use / scientific Python distributions
- List of basic tools (libraries) for science and their roles.
- Interfacing with other languages
- Language converters

Serial Lines (RS232) using pySerial

- Overview • Features • Requirements & Installation • Linux vs Windows
- Configuring & opening of Comm Ports • readline & testing ports
- pySerial API summary
- Examples of use

NumPy

- Data types • Extended precision
- 2-D Arrays • N-dimensional Arrays • Initializing / assigning arrays
- Operators on arrays • Array shape manipulation • I/O with NumPy
- Indexing • Broadcasting
- Structured array (records)
- Arithmetic series • Histograms • Randomization
- Grids • Indexing tricks

NumPy for MATLAB® users

- Key differences • array vs matrix • Indexing • Data types
- Short list of approximate MATLAB-NumPy Equivalents

Data Visualization – Matplotlib & misc

- Short introduction to Matplotlib
- Figures and Axes • non-linear axes
- Legends • Texts & annotating
- Simple 2-D plots practical examples
- Introduction to portable graphical Toolkits : PyQt and wxPython!

Ipython for interactive use and its future : Jupyter

- Use cases • Installation
- Symbolic calculation using SymPy

Signal Processing using SciPy

- Introduction to SciPy.signal
- Filtering • Convolution & correlation
- Introduction to Filter design • Spectral Analysis
- Example of FIR filter design