Présentation du langage Python

Cet article date un peu. En attendant une mise à jour (en cours), je vous recommande de consulter les sites suivants:

- Programmation-Python.org
- blogs.nuxeo.com (blogs sur Python et Zope)

Introduction

Python est un langage portable, dynamique, extensible, gratuit, qui permet (sans l'imposer) une approche modulaire et orientée objet de la programmation. Python est developpé depuis 1989 par Guido van Rossum et de nombreux contributeurs bénévoles.

Caractéristiques du langage

Détaillons un peu les principales caractéristiques de Python, plus précisément, du langage et de ses deux implantations actuelles:

- Python est **portable**, non seulement sur les différentes variantes d'UNiX, mais aussi sur les OS propriétaires: MacOS, BeOS, NeXTStep, M$-DOS et les différentes variantes de Window$. Un nouveau compilateur, baptisé JPython, est écrit en Java et génère du *bytecode* Java.
- Python est **gratuit**, mais on peut l'utiliser sans restriction dans des projets commerciaux.
- Python convient aussi bien à des **scripts** d'une dizaine de lignes qu'à des **projets complexes** de plusieurs dizaines de milliers de lignes.
- La **syntaxe** de Python est très simple et, combinée à des **types de données évolués** (listes, dictionnaires,...), conduit à des programmes à la fois très compacts et très lisibles. A fonctionnalités égales, un programme Python (abondament commenté et présenté selon les canons standards) est souvent de 3 à 5 fois plus court qu'un programme C ou C++ (ou même Java) équivalent, ce qui représente en général un temps de développement de 5 à 10 fois plus court et une facilité de maintenance largement accrue.
- Python gère ses ressources (mémoire, descripteurs de fichiers...) sans intervention du programmeur, par un mécanisme de **comptage de références** (proche, mais différent, d'un *garbage collector*).
- Il n'y a **pas de pointeurs** explicites en Python.
- Python est (optionnellement) **multi-threadé**.
- Python est **orienté-objet**. Il supporte l'**héritage multiple** et la **surcharge des opérateurs**. Dans son modèle objets, et en reprenant la terminologie de C++, toutes les méthodes sont virtuelle.

- Python intègre, comme Java ou les versions récentes de C++, un système d'**exceptions**, qui permettent de simplifier considérablement la gestion des erreurs.
- Python est **dynamique** (l'interpréteur peut évaluer des chaînes de caractères représentant des expressions ou des instructions Python), **orthogonal** (un petit nombre de concepts suffit à engendrer des constructions très riches), **reflectif** (il supporte la *métaprogrammation*, par exemple la capacité pour un objet de se rajouter ou de s'enlever des attributs ou des méthodes, ou même de changer de classe en cours d'exécution) et **introspectif** (un grand nombre d'outils de développement, comme le *debugger* ou le *profiler*, sont implantés en Python lui-même).
- Comme *Scheme* ou *SmallTalk*, Python est **dynamiquement typé**. Tout objet manipulable par le programmeur possède un type bien définit à l'exécution, qui n'a pas besoin d'être déclaré à l'avance.
- Python possède actuellement deux implémentations. L'une, **interprétée**, dans laquelle les programmes Python sont compilés en instructions portables, puis exécutés par une machine virtuelle (comme pour Java, avec une différence importante: Java étant statiquement typé, il est beaucoup plus facile d'accélérer l'exécution d'un programme Java que d'un programme Python). L'autre, génère directement du *bytecode* Java.
- Python est **extensible**: comme *Tcl* ou *Guile*, on peut facilement l'interfacer avec des librairies C existantes. On peut aussi s'en servir comme d'un **langage d'extension** pour des systèmes logiciels complexes.
- La **librairie standard** de Python, et les paquetages contribués, donnent accès à une grande variété de services: chaînes de caractères et expressions régulières, services UNIX standard (fichiers, *pipes*, signaux, sockets, threads...), protocoles Internet (Web, News, FTP, CGI, HTML...), persistence et bases de données, interfaces graphiques.
- Python est un langage qui **continue à évoluer**, soutenu par une communauté d'utilisateurs enthousisates et responsables, dont la plupart sont des supporters du logiciel libre. Parallèlement à l'interpréteur principal, écrit en C et maintenu par le créateur du langage, un deuxième interpréteur, écrit en Java, est en cours de développement.

Domaines d'application

Les domaines d'application naturels de Python incluent entre autres:

- L'apprentissage de la programmation objet.
- Les scripts d'administration système ou d'analyse de fichiers textuels.
- Tous les développement liés à l'Internet et en particulier au Web: scripts CGI, navigateurs Web, moteurs de recherche, agents intelligents, objets distribués...
- L'accès aux bases de données (relationnelles).
- La réalisation d'interfaces graphiques utilisateurs.

- Le calcul scientifique et l'imagerie. Python ne sert alors pas à écrire les algorithmes, mais à combiner et mettre en oeuvre rapidement des librairies de calcul écrites en langage compilé (C, C++, Fortran, Ada,...).
- Le prototypage rapide d'applications. L'idée générale est de commencer par écrire une application en Python, de la tester (ou de la faire tester par le client pour d'éventuelles modifications du cahier des charges). Trois cas peuvent alors se présenter:
    - Les performances sont satisfaisantes, après optimisation éventuelle du code Python. On livre alors le produit tel quel au client.
    - Les performances ne sont pas satifaisantes, mais l'analyse de l'exécution du programme (à l'aide du *profiler* de Python) montre que l'essentiel du temps d'exécution se passe dans une petite partie du programme. Les fonctions, ou les types de données, correspondants sont alors réécrits en C ou en C++, sans modification du reste du programme.
    - Sinon, il est toujours possible de réécrire tout le programme, en utilisant la version Python comme un brouillon.

    Même dans le pire des trois cas, il est très vraisemblable que le temps de développement aura été sensiblement plus court que si le programme avait été développé directement en C ou en C++.

Voici une liste de projets représentatifs basés sur Python:

- Zope, un serveur d'application innovant, et CPS, un framework de gestion de contenu et de travail collaboratif basé sur Zope.
- Les programmes d'administration système spécifiques à la distribution Red Hat Linux.
- Des moteurs de recherche comme Google ou Yahoo!.
- Chandler, le projet de PIM (Personnal Information Manager) de l'Open Source Applications Foundation

Références

Le site officiel de Python est www.python.org. On y trouvera la distribution officielle, de nombreux paquetages contribués, les compte-rendus des six conférences Python qui se sont déjà tenue à ce jour.

En plus de la documentation intégrée à la distribution standard, disponible également sur le site officiel et composée d'un tutoriel, du manuel de référence du langage et de celui de la librairie standard, on pourra consulter les deux livres parus actuellement sur Python (en anglais, pas de traduction prévue en français pour l'instant):

- *Programming Python*, de Mark Lutz, O'Reilly, 1996.

- *Internet Programming with Python (IPwP)*, de Aaron Watters, Guido van Rossum et James C. Ahlstrom, M&T Books, 1996.

*Une version éditée de cet article est parue dans le magazine **Programmez!** en novembre 1999.*