Les deux techniques de production des programmes

La compilation est la traduction du code source en langage objet. Elle comprend au moins quatre phases (trois phases d'analyse — lexicale, syntaxique et sémantique — et une phase de production de code objet). Pour générer le langage machine il faut encore une phase particulière : l'édition de liens. La compilation est contraignante mais offre au final une grande vitesse d'exécution.

Dans la technique de l'interprétation chaque ligne du code source analysé est traduite au fur et à mesure en instructions directement exécutées. Aucun programme objet n'est généré. Cette technique est très souple mais les codes générés sont peu performants : l'interpréteur doit être utilisé à chaque exécution...

Technique de production de Python

- Technique mixte : l'interprétation du bytecode compilé. Bon compromis entre la facilité de développement et la rapidité d'exécution ;
- le *bytecode* (forme intermédiaire) est portable sur tout ordinateur muni de la machine virtuelle Python.

Pour exécuter un programme, Python charge le fichier source .py (ou .pyw) en mémoire vive, en fait l'analyse syntaxique, produit le bytecode et enfin l'exécute. Pour chaque module importé par le programme, Python vérifie d'abord s'il existe une version précompilée du bytecode (dans un fichier .pyo ou .pyc) dont la date correspond au fichier .py. S'il y en a un, Python l'utilise, sinon il fait une analyse syntaxique du module .py, et utilise le bytecode qu'il vient de générer.

En pratique, il n'est pas nécessaire de compiler explicitement un module, Python gère ce mécanisme de façon transparente.

La construction des programmes

Le logiciel étudie les méthodes de construction des programmes. Plusieurs modèles sont envisageables, entre autres :

- la méthodologie procédurale. On emploie l'analyse descendante (division des problèmes) et remontante (réutilisation d'un maximum de sous algorithmes). On s'efforce ainsi de décomposer un problème complexe en sous-programmes plus simples. Ce modèle structure d'abord les actions ;
- la méthodologie objet. On conçoit des fabriques (*classes*) qui servent à produire des composants (*objets*) qui contiennent des données (*attributs*) et des actions (*méthodes*).

Les classes dérivent (*héritage* et *polymorphisme*) de classes de base dans une construction hiérarchique.
Python offre les *deux* techniques.

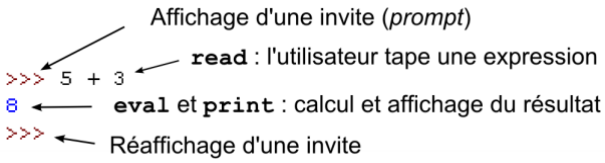
La présentation des programmes

Un programme *source* est destiné à l'être humain. Pour en faciliter la lecture, il doit être judicieusement *commenté*.

La signification de parties non triviales (et uniquement celles-ci) doit être expliquée par un commentaire. Un commentaire commence par le caractère # et s'étend jusqu'à la fin de la ligne.

Les deux modes d'exécution d'un code Python

- Soit on enregistre un ensemble de commandes Python dans un fichier grâce à un éditeur (on parle alors d'un *script Python*) que l'on exécute par une commande ou par une touche du menu de l'éditeur ;
- Soit on utilise un interpréteur (par exemple IDLE) pour obtenir un résultat immédiat grâce à l'interpréteur Python embarqué dans IDLE qui exécute la *boucle d'évaluation*



Les identifiants

Comme tout langage, Python utilise des *identifiants* pour nommer ses objets.

Définition

Un identifiant Python est une suite non vide de caractères, de longueur quelconque, formée d'un caractère de début et de zéro ou plusieurs caractères de continuation.

Sachant que :

- un caractère de début peut être n'importe quelle lettre Unicode, ainsi que le caractère souligné (_) ;
- un caractère de continuation est un caractère de début, un chiffre ou un point.

Attention

Les identifiants sont sensibles à la casse et ne doivent pas être un mot clé.