Sython Documentation

Version INDEV

LavaPower

oct. 02, 2018

Informations

Bienvenue sur la documentation du langage Sython.

Vous pouvez accéder rapidement aux différences informations via la barre de gauche via le sommaire de la page. Je vous conseille daller voir la page « introduction » si vous decouvrez Sython

---

Note : Il est important de ce rappeler que Sython est un projet OpenSource et développé par des personnes non professionnel.

Vous pouvez, vous aussi, y participer via le github.

Sommaire :

CHAPITRE 1

Introduction

Sython a été crée par LavaPower. Celui-ci voulait améliorer ces compétences en Python et décida de créer Sython, un langage simple mais demandant d'être rigoureux.

Sython est exempté de concept difficile et d'une syntaxe symbolique. Le but étant d'être langage permettant de se familiariser avec le monde de la programmation.

Note : « Langage simple » n'induit pas « Langage limité ». (Même si c'est le cas pour l'instant)

De plus il est important de comprendre que Sython va vous apprendre à être rigoureux dans vos syntaxes. Dans le sens où :

```
int a = 1 + 2
show(a)
```

Fonctionnera

```
int a = 1+2
show(a)
```

Fonctionnera aussi mais :

```
int a = 1+ 2
show(a)
```

Ne fonctionnera pas (soit vous aurez une erreur, soit un comportement non prévu)

CHAPITRE 2

FAQ

2.1 Qu'est-ce que Sython ?

Sython est un langage de programmation créé par LavaPower.

2.2 Pourquoi créer un nouveau langage de programmation ?

Le but premier n'est clairement pas de faire de la concurrence à des langages comme Python.

Le but en créant ceci est un entrainement à la programmation et un premier essai dans un domaine inconnu.

2.3 Quel sont les dépendances de Sython ?

Sython n'a besoin que de Python pour fonctionner.

2.4 Quels sont les platformes où Sython est utilisable ?

Du moment que Python est installable, Sython est utilisable. Cela inclus :

- Windows
- Linux
- Mac
- Et bien d'autres...

2.5 Quels logiciels puis-je utiliser pour coder en Sython ?

Actuellement, aucun IDE n'est compatible avec Sython...

Le mieux reste d'utiliser un éditeur de texte lambda comme Sublime Text ou encore Nodepad++

2.6 Je souhaite participer au développement de Sython, comment faire ?

Envoyez moi un message par Discord (LavaPower#1389) pour voir ce que vous pouvez faire

CHAPITRE 3

Téléchargement

3.1 Dépendances

Avant de télécharger Sython, il faut télécharger Python. Je vous invite à aller sur le site officiel : http://python.org et de prendre la dernière version.

Ensuite, pour avoir Sython, il faut télécharger la dernière release (ou le github pour la version en développement).

3.2 Dernière Release

PAS ENCORE SORTIE

3.3 Github

Github de Sython : http://github.com/LavaPower/Sython

3.4 Changelog

V 0.0.1 : Simplex Update (INDEV) :
- Première version

CHAPITRE 4

Installation

Aucune installation n'est à faire.

Pour utiliser Sython, il suffit de faire : "python Sython.py" <- Dans ce cas vous serez en mode "interactif", en mode console

Sinon, il faut créer votre programme avec comme extension ".sy" puis de lancer le programme via : "python Sython.py script.sy" <- Dans ce cas vous serez en mode "non-intéractif", en mode script

CHAPITRE 5

Variables

Sython est un langage fortement type ainsi vous connaissez toujours le type de votre variable. De plus, les conversions ne sont pas implicite.

5.1 Déclaration

Actuellement, Sython comporte 3 types basiques différents :
- int, pour les entiers
- float, pour les nombres à virgules
- str, pour les chaines de caractères

Pour déclarer une variable, il faut suivre le paterne suivant : <type> <nom> = <valeur>

Exemple :

```
int entier = 1
float flottant = 1.0
str texte = "Bonjour"
```

Note : Comme vous avez pû l'apercevoir, les flottants n'utilise pas une virgule mais un point pour différencier la partie entière de la partie décimal

5.2 Affectation

Si vous voulez réaffecter une nouvelle valeur à votre variable, vous pouvez la redéclarer comme au-dessus mais il est plus économique d'utiliser le paterne suivant : <nom> = <valeur>

Exemple :

```
// Déclaration
int entier = 1
//Nouvelle affection
entier = 2
```

Note : Ici, il y a aussi l'introduction des commentaires via le double symbole « // ».

Attention, les commentaires doivent être au début de la ligne et sans indentation.

5.3 Opérations

Actuellement, Sython supporte 4 opérations :

- Addition : "+"
- Soustraction : "+"
- Multiplication : "*"
- Division : "/"

Ces opérations sont utilisables dans la déclaration et dans l'affection. Cependant, elles sont la cible de restriction :

- Il est impossible d'appliquer une opération sur deux valeurs de type différent (Pas de conversion implicite)
- Il est impossible de multiplier, diviser, soustraire deux chaines de caractères. Seul l'addition est possible avec ce type.
- Le resultat d'une division entre deux entiers n'est pas forcément un entier et peut entrainer une perte de données s'il est enregistré dans un entier

Exemple :

```
int entier1 = 2
int entier2 = 3 + 4
int entier3 = entier1 * entier2
str phrase = "Bonjour "+"tout le monde"
```

5.4 Conversion

Malgré le fait que les conversions ne peuvent être fait par Sython, vous pouvez les faire par vous même. La technique est tout simplement une déclaration avec le nouveau type.

Exemple :

```
str nombre = "1"
// 'nombre' contient "1"
int nombre = nombre
// 'nombre' contient 1
```

Note : Attention, si la conversion n'est pas possible, vous aurez une erreur

CHAPITRE 6

Entrée - Sortie

C'est bien de créer des variables mais il est mieux d'afficher quelque chose nan ?

6.1 Affichage

L'affichage est très simple : il utilise la fonction "show()" qui prend un seul paramètre.

Exemples :

```
//Le bon vieux 'Hello World'
show("Hello World")
```

```
//Ce code va calculer le resultat de 2 + 2 puis l'afficher
int result = 2 + 2
//Ici la conversion est nécessaire pour additionner deux str entre eux
str result = result
show("Le résultat de 2 + 2 est "+result)
```

Note : Comme vous avez pû le voir, les additions (et autres opérations) sont possibles dans la fonction "show()"

6.2 Interaction

Afficher des trucs est sympa mais si nous voulons récupérer des informations écrites par l'utilisateur ?

C'est simple, la fonction "enter()" est faite pour vous ! Celle ci ne prend qu'un seul paramètre.

Exemples :

```
str name = enter("Entrez votre nom : ")
show("Votre nom est "+name)
```

```
int age = enter("Entrez votre age : ")
str age = age
show("Vous avez "+age+" ans")
```

Note : Dans notre deuxième code, la variable “age” est d’abord un int. Ceci permet que l’utilisateur ne rendre que des entiers, s’il ne le fait pas, le programmation crashera.

Il est prévu de rajouter des fonctions pour vérifier si un str peut être convertit en entier / flottant.

CHAPITRE 7

Conditions

7.1 Tests

Avant de créer des conditions avec des “if”... Il faut d’abord créer des tests.

Les tests sont des objets dont on veut savoir s’il est vrai ou pas.

En Sython, ces tests peuvent être :

- Une simple valeur / variable -> Il suffit de mettre la valeur ou le nom de la variable
- Une égalité entre deux valeurs -> If suffit de suivre ce paterne : “<valeur/variable> == <valeur/variable>”
- Une inégalité entre valeurs -> Il suffit d’utiliser des signes “<=” “>=” “<” “>” dans le parterne du haut à la place de “==”

7.2 If

Commençons par la condition la plus simple : “si <...> alors <...>”. En Sython, la synthaxe se démarque de Python avec l’utilisation d’accolade :

```
if <test> {
    <code>
}
```

Note : Le placement des accolades est à respecter, de plus, l’indentation ne peut se faire que via des espaces pour l’instant.

Exemple :

```
int a = 1
if a == 1 {
    show("a = 1")
}
```

Note : Ce code affichera “a = 1”

7.3 Elif

En complémentarité du “if”, ce trouve “elif”. Celui-ci correspond en français à “sinon si <...> alors <...>” et s'utilise comme le if.

```
if <test> {
    <code>
}
elif <test> {
    <code>
}
```

Exemple :

```
int age = enter("Entrez votre age : ")
if age > 18 {
    show("Vous êtes majeur depuis au moins un an !")
}
elif age == 18 {
    show("Vous êtes majeur depuis moins d'une année ! Bravo à vous !")
}
```

Note : Attention : Dans ce code, si l'utilisateur entre un nombre inférieur à 18, rien ne s'affichera mais nous allons y remédier.

7.4 Else

Le dernier du trio gagnant est le “else”. Il correspond à “sinon” et lui n'a pas de test, il sera vrai si les tests du haut sont faux.

```
if <test> {
    <code>
}
elif <test> {
    <code>
}
else {
    <code>
}
```

Exemple :

```
int age = enter("Entrez votre age : ")
if age > 18 {
    show("Vous êtes majeur depuis au moins un an !")
}
elif age == 18 {
```

(suite sur la page suivante)

(suite de la page précédente)

```
    show("Vous êtes majeur depuis moins d'une année ! Bravo à vous !")
}
else {
    show("Vous êtes mineur, profitez !")
}
```

CHAPITRE 8

Boucles

Les boucles ne sont pas encore codées dans le langage mais prévues dans un futur proche !

CHAPITRE 9

Fonctions

Les fonctions ne sont pas encore codées dans le langage mais prévues dans un futur proche !