Sython Documentation

Release INDEV

LavaPower

Apr 07, 2019

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 :

CHAPTER 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)

CHAPTER 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#2480) pour voir ce que vous pouvez faire

CHAPTER 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/sython-lang/Sython-V2

3.4 Changelog

V 0.0.1 : Initial Update (INDEV) :

- Première version

CHAPTER 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

CHAPTER 5

Variables

Sython est un langage où il n'est pas important de préciser le type de la variable. De plus, les conversions sont, dans la plupart des cas, implicite.

5.1 Déclaration

Actuellement, Sython comporte 4 types basiques différents :

- integer, pour les entiers
- float, pour les nombres à virgules
- string, pour les chaines de caractères
- boolean, soit vrai (true), soit faux (false)

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

Exemple :

```
entier = 1
flottant = 1.0
texte = "Bonjour"
etat = true
```

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 :

Exemple :

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

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

5.3 Opérations

Actuellement, Sython supporte 7 opérations :

- Addition : '+'
- Soustraction : '-'
- Multiplication : '*'
- Division : '/'
- Modulo : '%'
- Puissance : '^'
- Division Euclidienne : '//'

Exemple :

```
entier1 = 2
entier2 = 3 + 4
entier3 = entier1 * entier2
phrase = "Bonjour "+"tout le monde"
entier4 += entier2
```

Note: Ici, il y a aussi l'introduction des opérateurs affectifs via la notation : <variable> <operateur>= <valeur|variable> (Dispo pour tous les opérateurs).

5.4 Conversion

Malgré le fait que les conversions peuvent être fait par Sython, vous pouvez les faire par vous même :

Exemple :

```
nombre = "1"
// 'nombre' contient "1"
nombre = int(nombre)
// 'nombre' contient 1
```

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

Cependant, vous pouvez savoir si un conversion est possible via la fonction canbe qui s'utilise comme ceci 'canbe <variable> <type>'

Exemple :

```
nombre = "1"
lettre = "a"
show(canbe(nombre, "int")) #Affichera True
show(canbe(lettre, "int")) #affichera False
```

Note: Attention, les types sont en format contracté. Donc "integer" = "int", "string" = "str", "float" = "float" et "boolean" = "bool".

CHAPTER 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
result = 2 + 2
//La conversion est implicite
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 :

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

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

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

CHAPTER 7

Conditions

Pour contruire vos conditions, Sython vous propose 3 types d'outils : les comparateurs, les opérateurs logiques et les conditions en elle-même.

7.1 Comparateurs

Sython incorpore 5 comparateurs :

- Egal, noté "=="
- Inférieur, noté "<"
- Supérieur, noté ">"
- Inférieur ou Egal, noté "<="
- Supérieur ou Egal, noté ">="

Exemple :

```
a = "1"
b = 1
c = type(a) == type(b) # Sera égal à Faux vu que a et b ont pas le même type.
```

7.2 Opérateurs Logiques

Sython incorpore 3 opérateurs logiques :

- Et, noté "and" ou "&&"
- Ou, noté "or" ou "||"
- Non, noté "not" ou "!"

Exemple :

```
a = 18
b = 20
c = a == 18 && b == 20 # Sera égal à True
```

Note: Ici, vous pouvez remplacer && par and.

7.3 Conditions

Sython n'incorpore 4 types de conditions :

- If, noté 'if <condition> { <code> }'
- If-Else, noté 'if <condition> { <code> } else { <code> }'
- If-ElseIf, noté 'if <condition> { <code> } else if <condition> { <code> }' ou 'if <condition> { <code> } elseif <condition> { <code> }'
- If-ElseIf-Else, noté 'if <condition> { <code> } else if <condition> { <code> } else { <code> }' ou 'if <condition> { <code> } elseif <condition> { <code> } else { <code> }'

Exemple :

```
a = enter("Entrez un nombre entre 1 et 3 :")
a = int(a)
if a == 1
{
    show("Process 1")
}else if a == 2
{
    show("Process 2")
}elseif a == 3   # Ecrire else if ou elseif n'a pas d'importance.
{
    show("Process 3")
}else
{
    show("Erreur : Votre nombre n'est pas entre 1 et 3.")
}
```

Note: ATTENTION : Il ne doit pas y avoir de retour à la ligne entre } et else et entre } et else if. Sinon vous aurez une erreur.

CHAPTER 8

Boucles

Sython n'incorpore pour l'instant deux types de boucle. Mais on prévoit en plus la boucle for.

8.1 Loop

Cette boucle permet d'exécuter x fois une action. Sa syntaxe est 'loop <nombre> { <code> }'

Exemple :

```
a = "Bonjour"
loop 5
{
    show(a)
} # Affichera 5 fois "Bonjour"
```

8.2 While

Cette boucle permet d'exécuter une action tant qu'une expression est vraie. Sa syxtaxe est 'while <condition> { <code> }'

Exemple :

```
a = 0
while a < 10
{
    a ++
    show(a)
} # Affichera les nombres de 1 à 10 (bornes comprises)
```