KandyScript

Versión: 0.2 (beta)

Autor: Medina Dylan

KandyScript es un lenguaje de programación de soporte diseñado en Python. El mismo puede ser utilizado principalmente con dos fines:

1. Aprender programación en un ambiente donde se puede aprovechar la gran cantidad de herramientas que el lenguaje brinda, por ejemplo, la gran cantidad de estructuras de control o el uso de variables estáticas y dinámicas.
2. Aplicar este lenguaje en programas más grandes, centrando el uso del mismo en pequeñas tareas.

Notas sobre KandyScript:

* Es un lenguaje creado en Python, lo que le permite a usted reutilizar la gran mayoría de objetos y módulos que Python brinda.
* Si bien muchas estructuras funcionan de manera idéntica que en Python, hay algunas que son exclusivas de KandyScript.
* Es un lenguaje de programación orientado al aprendizaje y las pequeñas tareas debido a que su velocidad es reducida.

A lo largo del manual se irá comparando KandyScript con otros lenguajes de programación, mayormente Python.

Índice:

1. INTRODUCCIÓN A KANDYSCRIPT **4**

**1. INTRODUCCIÓN A KANDYSCRIPT:**

*KandyScript es un lenguaje interpretado que se ejecuta sobre Python, por tal motivo, la velocidad de KandyScript siempre será menor que la de Python.*

Comencemos con lo básico de KandyScript, nombrando algunas de sus ventajas y desventajas:

Ventajas:

* Posee una gran variedad de características, entre ellas están la mayoría de herramientas de Python.
* Tiene variables estáticas y dinámicas.
* Tiene constantes.
* Permite la creación de funciones y procedimientos.
* Permite la creación de clases (tiene soporte para la herencia de objetos Python o KandyScript).
* Permite la importación de módulos (librerías) tanto de KandyScript como de Python.
* Posee una gran cantidad de estructuras de control.

Desventajas:

* Posee una menor velocidad de ejecución.
* Requiere Python para funcionar.
* No tiene todas las herramientas de Python, por ejemplo, no es posible crear funciones generadores con ‘**yield**’.
* No es posible importar solo una función, clase, constante o variable de un módulo (de KandyScript o Python)
* No posee intérprete interactivo en consola.

**1.1 PRIMER CÓDIGO:**

Para comenzar a entender como KandyScript funciona, hay que entender cómo funciona su estructura.

Primeras reglas:

1. Las sentencias opcionalmente pueden terminar en “;”.
2. Se recomienda utilizar una sentencia por línea.

Primer programa: “Hola mundo”



Como era de esperar, el primer programa es un “Hola mundo.”

Análisis línea a línea:

1) Las primeras dos líneas son consideradas comentario, son notas para el programador y no serán ejecutadas por el intérprete.

2) En la línea 4 se puede observar la función “**print**”. La misma se usa para mostrar mensajes por pantalla.

3) En la misma línea, se puede observar la creación de un texto.

Para crear un texto en KandyScript se pueden utilizar comillas simples (') o comillas dobles (") indistintamente, lo único que hay que tener en cuenta es que se deben cerras las cadenas de texto con el mismo carácter con que se inició.

Salida por pantalla del programa:



Para continuar con el tutorial, se utilizará un código un poquito más complejo que el anterior:



El programa anterior y el actual tienen la misma finalidad: mostrar por pantalla el mensaje “Hola mundo”.

Análisis línea a línea:

1) Las primeras líneas no cambiaron mucho, siguen siendo comentarios.

2) En la línea 4 se realiza la primera creación de una variable y se le asigna el valor “Hola mundo”.

Para la creación de variables dinámicas hay que colocar un identificador seguido de un símbolo “=” y por último un valor o expresión.

Un identificador es un nombre que se le da a una variable que se aloja en la memoria RAM del equipo.

Un identificador válido debe cumplir con lo siguiente:

* No puede empezar con un número.
* No puede contener espacios en el nombre.
* Solo se permiten letras, números, y el símbolo “\_”.
* Tener al menos un carácter.
* No puede ser una palabra reservada.

|  |  |
| --- | --- |
| Identificadores válidos | Identificadores inválidos. |
| x9 | 9x |
| nombre\_y\_apellido | nombre y apellido |
| edad\_juan | edad-juan |
| Dni | D.N.I. |

Más adelante se explicará con más detalle el método usado para el nombramiento de las variables, constantes, funciones, procedimientos y clases.

3) La línea 5 muestra el resultado por consola.



**1.2 VARIABLES:**

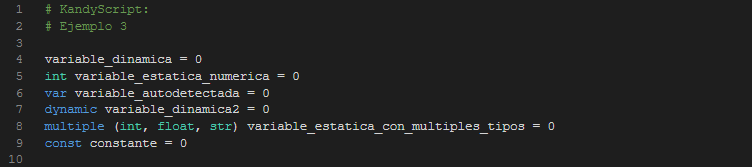
**Las variables:**

Las variables son un dato guardado en la memoria RAM del equipo.

En KandyScript estas variables pueden ser:

* Dinámicas: Pueden contener distintos valores y tipos, por ejemplo, una variable que puede guardar números o cadenas.
* Estáticas: Pueden contener distintos valores pero solo pueden almacenar variables con el tipo definido.
* Constantes: Solamente pueden almacenar un solo valor, y no pueden alterarse.

**Definir variables:**



Estructura para definir una variable dinámica:

1. identificador = valor
2. **dynamic** identificador = valor
3. **var** identificador = **None**
4. **var** identificador**;**

Estructura para definir una variable estática:

1. tipo identificador = valor
2. **var** identificador = valor
3. **const** identificador = valor

Tenga en cuenta que si bien la opción “b” dice que acepta un valor en general, dicho valor no puede ser “None” o “Undefined”.

Estructura para definir una variable estática con multiples tipos:

1. **multiple (**tipo, tipo, **…)** identificador = valor

Estructura para definir una variable con “Undefined”:

KandyScript posee un valor “Undefined” usado únicamente cuando a una variable no se le da un valor al definirla. Dicho valor no puede ser asignado a una variable directamente, cuando sea necesario el intérprete lo hará automáticamente.

1. tipo identificador;
2. **var** identificador;
3. **const** identificador;

El caso C es posible, aunque no tiene ninguna utilidad, puesto que ya existe una constante que guarda dicho valor, y es la palabra reservada “Undefined”