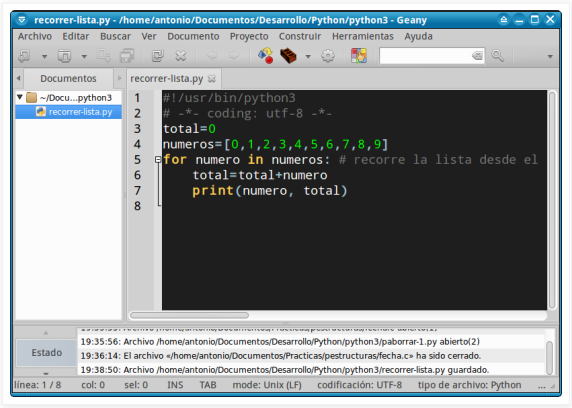
Python 3 para impacientes

"Simple es mejor que complejo" (Tim Peters)

miércoles, 29 de enero de 2014

Edición y ejecución de programas Python



Para escribir los programas se puede usar un editor como **Geany** que soporta codificación UTF-8, es liviano y multiplataforma.

Instalación y configuración del editor Geany

Si utiliza una distribución GNU/Linux como Ubuntu, para instalar el editor **Geany** utilizar el propio Centro de Software. Si tiene otra distribución o Windows puede descargar Geany desde la sección "**Download**" de la web de la aplicación www.geany.org y una vez descargada ejecutar el instalador.

Al iniciar Geany por primera vez se crea un documento vacío y sin nombre con la etiqueta "**sin título**". Para realizar el primer programa escribir la siguiente línea:

print("Python para impacientes")

A continuación, guardamos el documento con el nombre "python-001.py" con la opción "**Guardar**" del menú "**Archivo**" o con la combinación de teclas [**Ctrl+S**].

Una vez almacenado el archivo, el editor Geany, por la extensión que hemos indicado en su nombre, identifica que el código introducido es Python y aplica colores a la sintaxis:

```
print("Python para impacientes")
```

En adelante, la característica de aplicar color a la sintaxis seguirá funcionando.

Antes de ejecutar el primer programa es necesario comprobar que el intérprete que realizará dicha tarea es Python3 (y no una versión anterior). Para ello, accedemos a la opción "**Establecer comandos de construcción**" del menú "**Construir**" y verificamos que el comando del "**Compilador**" del apartado "**Comandos de Python**" y el comando "**Ejecutar**" del apartado "**Ejecutar comandos**" tengan los siguientes valores:

Buscar

Buscar

Python para impacientes

Python
IPython
EasyGUI
Tkinter
JupyterLab
Numpy

Anexos

Guía urgente de MySQL
Guía rápida de SQLite3

Entradas + populares

Dar color a las salidas en la consola
En Python para dar color a las salidas en la consola (o en la terminal de texto) existen varias posibilidades. Hay un método basado ...

Instalación de Python, paso a paso
Instalación de Python 3.6 A finales de 2016 se produjo el lanzamiento de Python 3.6 . El propósito de esta entrada es mostrar, pas...

Añadir, consultar, modificar y suprimir elementos en Numpy
Acceder a los elementos de un array. [], [,], ... Acceder a un elemento de un array. Para acceder a un elemento se utiliz...

Variables de control en Tkinter
Variables de control Las variables de control son objetos especiales que se asocian a los widgets para almacenar sus valore...

Cálculo con arrays Numpy
Numpy ofrece todo lo necesario para obtener un buen rendimiento cuando se trata de hacer cálculos con arrays. Por como está concebid...

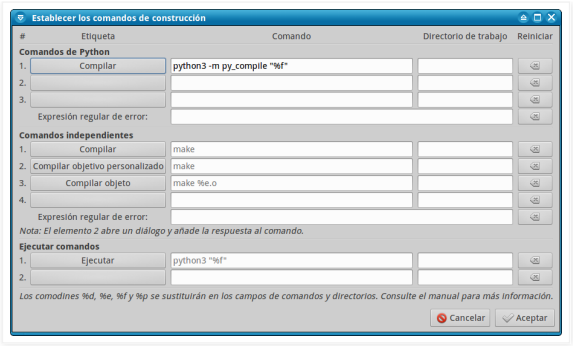
Tkinter: interfaces gráficas en Python
Introducción Con Python hay muchas posibilidades para programar una interfaz gráfica de usuario (GUI) pero Tkinter es fácil d...

Operaciones con fechas y horas. Calendarios
Los módulos datetime y calendar amplían las posibilidades del módulo time que provee funciones para manipular expresiones de ti...

Convertir, copiar, ordenar, unir y dividir arrays Numpy
Esta entrada trata sobre algunos métodos que se utilizan en Numpy para convertir listas en arrays y viceversa; para copiar arrays d...

Tkinter: Tipos de ventanas
Ventanas de aplicación y de diálogos En la entrada anterior tratamos los distintos gestores de geometría que se utilizan para di...

El módulo random
El módulo random de la librería estándar de Python incluye un conjunto de funciones



Comandos de Python:

Compilar: **python3 -m py_compile "%f"**

Ejecutar comandos:

Ejecutar: **python3 "%f"**

Si es preciso modificar y aceptar los cambios. Estos cambios serán permanentes, es decir, para nuevos programas se utilizará el intérprete de Python3.

Finalmente, ejecutamos el primer programa con la tecla [**F5**] o con la opción "**Ejecutar**" del menú "**Construir**", o bien, con el botón "**Ejecutar**" de la barra de herramientas.

El resultado será mostrado en una ventana del emulador de Terminal:

Python para impacientes

Con la tecla [**Enter**] regresar al editor.

El siguiente proyecto "python-002.py" es algo más ambicioso porque incluye después de la línea de "for..." un bloque de líneas sangradas. Recomendamos teclear el código (no copiarlo) para comprobar como **Geany** sangra automáticamente cuando el guion lo requiere.

```
#!/usr/bin/python3
# -*- coding: utf-8 -*-
total=0
numeros=[0,1,2,3,4,5,6,7,8,9]
for numero in numeros: # recorre lista desde primer elemento a último
    total=total+numero
    print(numero, total)
```

Después de guardar el programa "python-002.py", ejecutarlo con [**F5**]. El resultado será una lista de números desde el 0 al 9 y otra con la suma acumulada.

Por ahora no necesitamos saber más sobre **Geany**. En capítulos posteriores iremos aprendiendo más sobre su funcionamiento.

Convertir un programa en ejecutable

Incluir la ruta del intérprete Python en los programas:

En la primera línea de un programa debemos incluir la ruta donde está instalado el intérprete Python3 que deseamos invocar, como en el programa anterior. Ejemplos:

#!/usr/bin/python3
#!/usr/bin/env python3

La primera ruta es la habitual en un sistema GNU/Linux. La segunda ruta suele utilizarse en los programas que van a ejecutarse en distintos sistemas. Si tenemos instalado Python3 con el comando "**which python3**" podemos conocer su ruta.

Convertir en ejecutable y ejecutar un programa desde la línea de comandos:

$ chmod +x programa.py
$ ./programa.py

Añadir al PATH del sistema la ruta de un programa:

$ export PATH=$PATH:/home/carpetaprograma

Codificación (encoding)

que permiten obtener de distintos modos números a...

Archivo

enero 2014 (10)

python.org

pypi.org

Sitios

- ActivePython
- Anaconda
- Bpython
- Django
- Flask
- Ipython
- IronPython
- Matplotlib
- MicroPython
- Numpy
- Pandas
- Pillow
- PortablePython
- PyBrain
- PyCharm
- PyDev
- PyGame
- Pypi
- PyPy
- Pyramid
- Python.org
- PyTorch
- SciPy.org
- Spyder
- Tensorflow
- TurboGears

Para que el intérprete Python3 reconozca además de los caracteres comunes de los distintos alfabetos, aquellos que son característicos de cada idioma (como sucede con la ñ en el castellano/español, con las vocales acentuadas o la diéresis) **si la codificación del archivo fuente no es UTF-8** tendremos que insertar en el comienzo de nuestros programas la siguiente línea:

```
# -*- coding: utf-8 -*-
```

En los fuentes sin codificación UTF-8 que olvidemos indicarla y escribamos, por ejemplo, vocales acentuadas o la ñ, el intérprete Python no será capaz de reconocer estos caracteres y producirá un error.

Incluir # Comentarios en el código

Además de los usos que hemos visto con anterioridad de la almohadilla "#", este carácter se utiliza específicamente para comentar líneas de programa:

```
# Esto es un comentario

if caracter in 'Python':  # Si carácter está en 'Python'
```

Las **cadenas de documentación** o **docstring** son comentarios que pueden abarcar varias líneas, que comienzan y terminan con tres comillas (simples o dobles) y se sitúan al principio de los módulos, de las funciones y las clases para explicar la finalidad que tienen:

```
"""Comentario extenso
"""
```

Relacionado:

- Docstrings
- Programas con estilo en Python

Ir al índice del tutorial de Python

Publicado por Pherkad en 12:17

Etiquetas: Python3

Entrada más reciente Inicio Entrada antigua