

# SHELL SCRIPTING: ANÁLISIS DE ARGUMENTOS ENVIADOS POR LÍNEA DE COMANDOS MEDIANTE PYTHON CON ARGPARSE

ARGPARSE ES UN MÓDULO DE LA LIBRERÍA ESTÁNDAR DE PYTHON, QUE REEMPLAZANDO A OPTPARSE DESDE LA VERSIÓN 2.7 DEL LENGUAJE, SE HA CONVERTIDO EN EL MÓDULO POR EXCELENCIA PARA ANALIZAR LOS ARGUMENTOS ENVIADOS A TRAVÉS DE LA LÍNEA DE COMANDOS.

El módulo `argparse` forma parte de la librería de módulos estándar de Python y su finalidad es la de analizar los argumentos enviados al programa mediante línea de comandos, facilitando las mismas funcionalidades que el obsoleto `optparse` pero incorporando ciertas características con las que éste no contaba.

Se trata de un módulo muy simple de utilizar y no necesariamente será implementado solo por especialistas en Python: también es una excelente alternativa para crear de forma rápida un *script* principal (*main*) para cualquier tipo de aplicación de consola, incluso, aquellas programadas con `bash` u otros lenguajes que no cuenten con tanta facilidad para, por ejemplo, el análisis de argumentos y/o la generación de ayudas en pantalla.

Con solo agregar una lista de argumentos a `ArgumentParser()`, el módulo se encargará de poner a disposición del usuario de la aplicación, los argumentos `-h` y `--help` y generar de forma automática, textos de ayuda similares al siguiente:

```
eugenia@cococha-gnucita:~/HDMagazine/12$ python newhost --help
usage: newhost [-h] [-v] -d DOMAIN [-a [ALIAS [ALIAS ...]]]
              [-l [{static,python,php}]] [-u USERNAME] [-p PATH]
              [-lp LOGPATH] [--send-email] [-e EMAIL]
```

Prepara el ambiente necesario para hospedar un nuevo dominio en Ubuntu Server 12.04 LTS o versiones posteriores

optional arguments:

```
-h, --help          show this help message and exit
-v, --version       show program's version number and exit
```

```
-d DOMAIN, --domain DOMAIN
    Nombre del dominio a configurar
-a [ALIAS [ALIAS ...]], --alias [ALIAS [ALIAS ...]]
    Alias de dominio
-l [{static,python,php}], --language [{static,python,php}]
    Lenguaje predeterminado del sitio Web
-u USERNAME, --user USERNAME
    Usuario del dominio
-p PATH, --path PATH
    Directorio raíz de archivos Web
-lp LOGPATH, --log-path LOGPATH
    Directorio en el que serán almacenados los logs de
    Apache
--send-email
    Si se indica, enviará un e-mail con los datos del
    nuevo dominio.
-e EMAIL, --email EMAIL
    Válido si --send-email se ha indicado.
```

Como se puede observar en el bloque anterior, dos argumentos por defecto, son descriptos al comienzo: `help` y `version`. Ambos son facilitados por `argparse` para mostrar la ayuda y versión del programa, respectivamente. Incluso, la ayuda de uso será mostrada si los argumentos recibidos no son los esperados:

```
eugenia@cococho-gnucita:~/HDMagazine/12$ python newhost
usage: newhost [-h] [-v] -d DOMAIN [-a [ALIAS [ALIAS ...]]]
              [-l [{static,python,php}]] [-u USERNAME] [-p PATH]
              [-lp LOGPATH] [--send-email] [-e EMAIL]
newhost: error: argument -d/--domain is required
```

Todo esto es lo que `argparse` pondrá a disposición del usuario, con unas pocas líneas de código fuente.

## INTRODUCCIÓN

### Importación del módulo:

Para comenzar a utilizar `argparse`, bastará con importar la clase `ArgumentParser()`:

```
from argparse import ArgumentParser
```

### Construcción de un objeto `ArgumentParser`:

Construir un objeto `ArgumentParser`, es una forma de inicializar los datos principales de la aplicación. El método constructor del objeto `ArgumentParser` (función `__init__`), si bien puede ser invocado sin argumentos, permite (entre otros), los siguientes parámetros:

<b>prog</b>	por defecto es el nombre del archivo aunque en circunstancias muy puntuales, podría modificarse
<b>description</b>	una descripción del programa que será mostrada al inicio de la ayuda
<b>epilog</b>	texto que será mostrado al final de la ayuda

**version**          número de versión del programa

Todos estos parámetros son opcionales pero sin embargo, si se indicase el parámetro `version`, automáticamente tendríamos de la opción `-v` y `-version`:

```
#!/usr/bin/env python
from argparse import ArgumentParser

argp = ArgumentParser(
    version='1.0',
    description='Descripción breve del programa',
    epilog='Copyright 2013 Autor bajo licencia GPL v3.0'
)
```

Otros **parámetros admitidos por `ArgumentParser.__init__()`** pueden verse en la siguiente URL del manual oficial: <http://docs.python.org/2/library/argparse.html#argumentparser-objects>

## AGREGANDO ARGUMENTOS CON `ARGUMENTPARSER.ADD_ARGUMENT`

Cuando se crea un objeto `ArgumentParser`, éste dispone de un método `add_argument()` que como su nombre lo indica, tiene por finalidad agregar argumentos de a uno por vez. Este método, puede recibir como parámetro, un nombre de argumento o una lista de banderas (*flags*). Por ejemplo:

```
argp.add_argument('directorio', '-d', '--directorio')
argp.add_argument('dominio')
argp.add_argument('-l')
argp.add_argument('--listar')
argp.add_argument('-p', '--printer')
```

Además del argumento en sí mismo, `add_argument()` puede recibir muchos otros parámetros. Entre los más frecuentes, podemos encontrar los siguientes:

### **action**

Descripción: Acción que se deberá realizar con el valor del argumento.

Valores posibles:

<code>store</code>	almacena el valor (acción predeterminada)
<code>store_const</code>	si el argumento es pasado, almacenará el valor definido en el parámetro <code>const</code> (ver más abajo). Es útil cuando se requiere recibir un flag pero sin valor asociado
<code>store_true / store_false</code>	Igual que <code>store_const</code> pero no necesita definir el valor de <code>const</code>

ya que almacenarán True o False respectivamente en caso que el argumento sea pasado

**append** almacena los valores del argumento en una lista. Es útil cuando un mismo argumento puede indicarse varias veces con diferentes valores. Por ejemplo: `--argumento valor1 --argumento valor2` generaría `argumento = ['valor1', 'valor2']`

**append\_const** almacena el valor de `const` en una lista. Especialmente útil cuando el valor de diferentes argumentos es una constante y se los necesita de forma unificada (ver ejemplo para mejor comprensión). Requiere que el parámetro `dest` (ver más abajo) posea el mismo valor en los diferentes argumentos

Valor por defecto: store

Obligatorio: NO

Ejemplo:

```
argp.add_argument('--table', action='store')
--table foo genera: table = 'foo'
argp.add_argument('--table', action='store_const', const='users')
--table genera: table = 'users'
argp.add_argument('--table', action='append')
--table foo --table bar genera: table = ['foo', 'bar']
argp.add_argument('--php', dest='lenguajes',
                  action='append_const', const='php')
argp.add_argument('--python', dest='lenguajes',
                  action='append_const', const='python')
--php --python genera: lenguajes = ['php', 'python']
```

## **nargs**

Descripción: Cantidad de valores que pueden recibirse para el argumento en cuestión.

Valores posibles: el literal de un entero (incluso cuando sea 1, retornará una lista), o sino:

? uno o ninguno

+ uno o más

\* cero o más

REMAINDER todos los argumentos se recogen en una lista.

Esto es especialmente útil, cuando los comandos recibidos se requieren pasar a otra aplicación.

Valor por defecto: uno solo

Obligatorio: NO

Ejemplo:

```
argp.add_argument('--table', nargs='+')  
argp.add_argument('--rango', nargs=2)
```

### default

Descripción: Un valor por defecto para el argumento.

Valores posibles: cualquiera

Valor por defecto: ninguno

Obligatorio: NO

Ejemplo:

```
argp.add_argument('--host', default='localhost')
```

### type

Descripción: El tipo de datos

Valores posibles: str, int, etc.

Valor por defecto: None

Obligatorio: NO

Ejemplo:

```
argp.add_argument('-n, --nombre', type=str)  
argp.add_argument('--edad', type=int)
```

### choices

Descripción: Una lista de opciones con valores posibles

Valores posibles: una lista

Valor por defecto: None

Obligatorio: NO

Ejemplo:

```
argp.add_argument('-l, --language', choices=['php', 'bash', 'ruby'])
```

### required

Descripción: Indica si el argumento es o no obligatorio

Valores posibles:

True	Argumento obligatorio
False	Argumento no obligatorio

Valor por defecto: False

Obligatorio: NO

Ejemplo:

```
argp.add_argument('--obligatorio', required=True)  
argp.add_argument('--opcional', required=False)
```

### help

Descripción: Texto a mostrar en la ayuda del argumento.

Valores posibles: cadena de texto

Valor por defecto: None

Obligatorio: NO (aunque es muy recomendado indicarlo)

Ejemplo:

```
argp.add_argument('--list', help='Retorna la lista de tablas en la DB')
```

### **dest**

Descripción: Nombre que se utilizará para la variable que almacenará el valor del argumento.

Valores posibles: string con nombre de variable válido

Valor por defecto: el nombre argumento o flag

Obligatorio: NO

Ejemplo:

```
argp.add_argument('-p', dest='path')
argp.add_argument('-h', dest='hostname')
```

## **GENERAR EL ANÁLISIS DE LOS ARGUMENTOS CON ARGUMENTPARSER.PARSE\_ARGS**

Finalmente, se necesitará indicar a ArgumentParser que analice los argumentos:

```
argumentos = argp.parse_args()
```

parse\_args retornará cada argumento indicado por línea de comandos, según su configuración, como propiedades del objeto generado:

```
argumentos = argp.parse_args()
suponiendo un argumento cuyo destino sea llamado foo, obtendríamos su valor con:
print argumentos.foo
```

A continuación, el código que generó el texto de ayuda del ejemplo al comienzo del artículo:

```
#!/usr/bin/env python
# -*- coding:utf-8 -*-
from argparse import ArgumentParser

argp = ArgumentParser(
    prog='newhost',
    description="""Prepara el ambiente necesario para hospedar un nuevo
dominio en Ubuntu Server 12.04 LTS o versiones posteriores""",
    epilog='Copyright 2013 Eugenia Bahit - GPL v3.0',
    version='New WebSite Hosting beta 1.0'
)

argp.add_argument('-d', '--domain', action='store', required=True,
    help='Nombre del dominio a configurar', dest='domain')
```

```
argp.add_argument( '-a', '--alias', action='store', required=False, nargs='*',
                  help='Alias de dominio', dest='alias' )

argp.add_argument( '-l', '--language', choices=['static', 'python', 'php'],
                  action='store', required=False, default='static', nargs='?',
                  help='Lenguaje predeterminado del sitio Web', dest='language' )

argp.add_argument( '-u', '--user', action='store', default='www-data', required=False,
                  help='Usuario del dominio', dest='username' )

argp.add_argument( '-p', '--path', action='store', default='/srv/websites/',
                  required=False, help='Directorio raíz de archivos Web', dest='path' )

argp.add_argument( '-lp', '--log-path', action='store', default='/srv/websites/logs/',
                  required=False,
                  help='Directorio en el que serán almacenados los logs de Apache',
                  dest='logpath' )

argp.add_argument( '--send-email', action='store_true', required=False,
                  help='Si se indica, enviará un e-mail con los datos del nuevo dominio.',
                  dest='sendemail' )

argp.add_argument( '-e', '--email', action='store', required=False,
                  help='Válido si --send-email se ha indicado.', dest='email' )

args = argp.parse_args()
print vars(args)

# ***** FIN CÓDIGO SCRIPT PYTHON *****
# ejecución del script

eugenia@cocochoa-gnucita:~/HDMagazine/12$ python newhost -v
New WebSite Hosting beta 1.0

eugenia@cocochoa-gnucita:~/HDMagazine/12$ python newhost -d eugeniabahit.com -a
www.eugeniabahit.com -l php -u juanito
{'username': 'juanito', 'domain': 'eugeniabahit.com', 'language': 'php', 'sendemail': False,
 'logpath': '/srv/websites/logs/', 'alias': ['www.eugeniabahit.com'], 'path':
 '/srv/websites/', 'email': None}
```



# PgDay Argentina 2013

14 de Noviembre - Buenos Aires

Inscribite en: [www.PgDay.com.ar](http://www.PgDay.com.ar)