

# Diseña tu Propia CLI con Python

The background features a light blue gradient. A large, dark blue arrow-shaped graphic points from the left towards the right, containing the main title. Below this, a horizontal orange bar is positioned, with a dark blue arrow-shaped graphic pointing from the right towards the left, overlapping the orange bar.

# Hola !

Quien soy :

Cesar A. Herdenez Araque

Técnico en Sistemas - SENA

Ing. en Sistemas - F.U. San Martin

Auxiliar Comercial en una Entidad Financiera

Backend Developer Python & Golang

Comenzando con Angular 4




[hello@cesarh.co](mailto:hello@cesarh.co)

<https://twitter.com/cesarau0619>

<https://github.com/Cesar0510/>

1

# QUE ES PYTHON



Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”.

Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible.

Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos.

2

**¿POR QUÉ USAR  
PYTHON ?**

# Who uses Python?



Confidence Comes Standard.™



CANONICAL



mozilla  
Firefox



**WHO'S AWESOME?**



**YOU'RE AWESOME!**

quickmeme.com

3

**QUE NECESITAMOS**





## Herramientas

### Editores de Texto y IDEs

<https://www.sublimetext.com/>

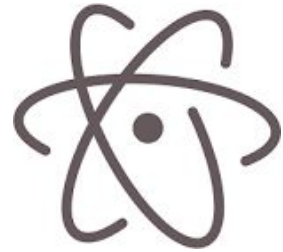
<https://atom.io/>

<https://code.visualstudio.com/>

<https://www.jetbrains.com/pycharm/>

<http://www.vim.org/>

<https://www.gnu.org/software/emacs/>





## Herramientas

Python

<https://github.com/pyenv/pyenv>

<https://conda.io/docs/>

<https://www.python.org/>

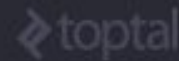
The logo for Conda, featuring a green circular icon with a white grid pattern on the left, followed by the word "CONDA" in a bold, green, sans-serif font.

The logo for Python, featuring the blue and yellow Python logo icon on the left, followed by the word "python" in a grey, lowercase, sans-serif font with a trademark symbol (TM) to the right.



## Herramientas

Implementación	Máquina Virtual	Ej) Lenguaje Compatible
CPython	CPython VM	C
Jython	JVM	Java
IronPython	CLR	C#
Brython	Motor Javascript(Por Ej., V8)	JavaScript
RubyPython	Ruby VM	Ruby



<https://www.toptal.com/python/por-que-hay-tantos-pythons/es>

4

**QUE VAMOS A VER**





## Veremos

- Tipos básicos
- Colecciones
- Control de flujo
- Funciones
- Orientación a Objetos
- Bucles
- Métodos especiales
- Decoradores

# Modulo argparse

```
class argparse.ArgumentParser(prog=None, usage=None, description=None, epilog=None, parents=[],  
formatter_class=argparse.HelpFormatter, prefix_chars='-', fromfile_prefix_chars=None, argument_default=None, conflict_handler='error',  
add_help=True, allow_abbrev=True)
```

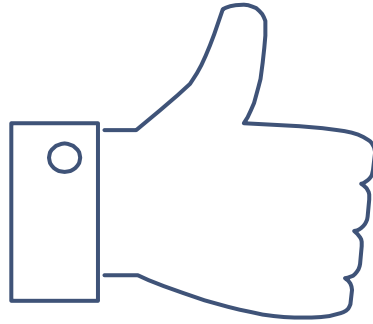
Create a new `ArgumentParser` object. All parameters should be passed as keyword arguments. Each parameter has its own more detailed description below, but in short they are:

- `prog` - The name of the program (default: `sys.argv[0]`)
- `usage` - The string describing the program usage (default: generated from arguments added to parser)
- `description` - Text to display before the argument help (default: none)
- `epilog` - Text to display after the argument help (default: none)
- `parents` - A list of `ArgumentParser` objects whose arguments should also be included
- `formatter_class` - A class for customizing the help output
- `prefix_chars` - The set of characters that prefix optional arguments (default: '-')
- `fromfile_prefix_chars` - The set of characters that prefix files from which additional arguments should be read (default: None)
- `argument_default` - The global default value for arguments (default: None)
- `conflict_handler` - The strategy for resolving conflicting optionals (usually unnecessary)
- `add_help` - Add a `-h/--help` option to the parser (default: True)
- `allow_abbrev` - Allows long options to be abbreviated if the abbreviation is unambiguous. (default: True)

## Modulo argparse

- **name or flags** - Either a name or a list of option strings, e.g. `foo` or `-f`, `--foo`.
- **action** - The basic type of action to be taken when this argument is encountered at the command line.
- **nargs** - The number of command-line arguments that should be consumed.
- **const** - A constant value required by some **action** and **nargs** selections.
- **default** - The value produced if the argument is absent from the command line.
- **type** - The type to which the command-line argument should be converted.
- **choices** - A container of the allowable values for the argument.
- **required** - Whether or not the command-line option may be omitted (optionals only).
- **help** - A brief description of what the argument does.
- **metavar** - A name for the argument in usage messages.
- **dest** - The name of the attribute to be added to the object returned by `parse_args()`.





# Gracias!

Preguntas ??

Pueden encontrarme en

[hello@cesarh.co](mailto:hello@cesarh.co)

<https://twitter.com/cesarau0619>