Eugenia Bahit

Curso: Python para Principiantes

www.eugeniabahit.com

Comparte el conocimiento

Eres libre de:

- **Copiar**, **distribuir** y **compartir** este libro

Bajo las siguientes condiciones:

- Reconocer y respetar la autoría de la obra
- No hacer uso comercial de ella
- No alterar el contenido

http://www.safecreative.org/work/1207302042960

Descarga todos los ejemplos de este libro y los talleres del curso, ingresando en

http://curso-python.eugeniabahit.com/sources.tar.gz

00 Tabla de Contenidos (índice)

01 Preparación del entorno de desarrollo

Comenzaremos instalando todo nuestro sistema, para crear un entorno de desarrollo propicio, para trabajar con Python. A tal fin, nos valdremos de las siguientes herramientas y tecnologías:

1. Sistema Operativo GNU/Linux: Ubuntu 11.10 (o superior)
2. Python 2.7
3. iPython (Shell interactivo mejorado)
4. Ninja-IDE (IDE de desarrollo)
5. Bazaar (Sistema de Control de Versiones distribuido)

Introducción al Sistema Operativo GNU/Linux

Antes de comenzar, intentaremos establecer una diferencia, entre los término “Linux” y “GNU/Linux”, a fin de saber de qué estamos hablando con exactitud, en cada caso.

Linux, es un kernel, es decir, el núcleo de un Sistema Operativo, mientras que **GNU/Linux, el Sistema Operativo que utiliza el Kernel Linux** como núcleo, creado, difundido y promovido a través del **Proyecto GNU**, por la **Free Software Foundation**, organización sin fines de lucro, fundada por Richard Stallman, principal precursor del Software Libre.

El Kernel Linux, parte fundamental del Sistema Operativo, fue desarrollado por **Linus Torvals**, utilizando como modelo a UNIX. Una de las diferencias fundamentales entre los núcleos Linux y UNIX, es que el primero, es Software Libre, mientras que el segundo no lo es.

Por otra parte, mientras existe un único Kernel Linux (con versiones diferentes), existen decenas y hasta cientos de **distribuciones GNU/Linux**, es decir, diferentes Sistemas Operativos basados en el Kernel Linux, entre las cuales se destacan: **Debian**, **Ubuntu**, **Kubuntu**, **Fedora**, **Gentoo**, **Slackware**, **CentOS**, **ArchLinux**, **Asturix**, entre otros cientos.

Más información al respecto, puede encontrarse en:

- Sitio Web de la **Free Software Foundation**: www.fsf.org
- Sitio Web del **Proyecto GNU**: www.gnu.org
- Sitio Web del **Kernel Linux**: http://www.kernel.org/
- Sitio Web de la **Linux Foundation**: http://www.linuxfoundation.org/
- Introduccion al software libre (Universitat Obierta de Catalunya)
- Sistema operativo gnu linux basico (Universitat Obierta de Catalunya)

Instalación de Ubuntu GNU/Linux en Windows

Si eres usuario de Windows y deseas conservar tu Sistema Operativo actual, **puedes descargar Ubuntu Windows Installer** desde el sitio Web oficial de Canonical (empresa que desarrolla y mantiene Ubuntu) en la siguiente URL: http://www.ubuntu.com/download/ubuntu/windows-installer

Ubuntu Windows Installer se instalará desde el propio MS Windows© como si fuese un Software más, permitiéndote iniciar tu ordenador con Ubuntu o MS Windows© según

elijas.

Para instalar Ubuntu Windows Installer, **sigue las instrucciones de los pasos 2 y 3 de la URL de descarga**, las cuales podrás visualizar pulsando el botón “Show me how” de cada uno de los pasos.

Instalación de Ubuntu GNU/Linux como único Sistema Operativo

Para instalar Ubuntu como único Sistema Operativo, sigue los siguientes pasos:

1. ingresa en http://www.ubuntu.com/download/ubuntu/download
2. En el paso 1, selecciona la versión de Ubuntu que deseas descargar. Para procesadores de un solo núcleo, selecciona la versión 10.04 LTS. Para procesadores más modernos, puedes seleccionar la última versión (versión que aparece seleccionada por defecto en el desplegable de versiones). Si tienes dudas sobre si elegir la versión para 32 o 64 bits, elige la de 32-bits. Pulsa el botón “Start download” y aguarda a que se descargue el archivo.
3. Una vez descargado el archivo, podrás quemarlo en un CD/DVD o un Pendrive USB. En el paso 2 de la URL de descarga, selecciona CD o USB stick según tus preferencias y el Sistema Operativo desde el cual harás la copia (Windows o Mac). Pulsa el botón “show me how” y sigue las instrucciones de quemado.
4. A continuación, salta al paso 4 del sitio de descarga (el 3 es solo para probar Ubuntu sin instalarlo); pulsa el botón “show me how” y sigue las instrucciones para instalar Ubuntu en tu ordenador.

Instalando Python

Una vez que hayas instalado tu distribución GNU/Linux, ya tendrás Python instalado en tu sistema.

Para comprobarlo, abres una terminal (presiona Alt + F4 y luego escribe en el campo de búsqueda gnome-terminal) y escribe python como se muestra a continuación:

```
eugenia@cochito:~$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Lo que verás en pantalla, es el Shell interactivo de Python. **Para salir del Shell interactivo, pulsa las teclas Ctrl + D.**

```
eugenia@cochito:~$ python
Python 2.7.2+ (default, Oct  4 2011, 20:03:08)
[GCC 4.6.1] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> print "Hola Mundo!"
Hola Mundo!
>>>
```

Si en lugar del Shell interactivo, ves un mensaje de error similar a “python: orden no encontrada”, deberás seguir los siguientes pasos para instalarlo:

Actualiza la lista de los repositorios:

```
eugenia@cochito:~$ sudo apt-get update
```

Actualiza el Sistema Operativo:

```
eugenia@cochito:~$ sudo apt-get upgrade
```

Instala Python:

```
eugenia@cochito:~$ sudo apt-get install python2.7
```

SOBRE LOS COMANDOS
sudo: te convierte en super usuario. Único usuario que tiene permisos para instalar paquetes en tu sistema operativo.
apt-get: es la utilidad para manejar paquetes en distribuciones GNU/Linux basadas en Debian. Alternativamente, puedes utilizar

el comando **aptitude** en vez de apt-get.
update: opción de apt-get que sincroniza los archivos del índice de paquetes con los repositorios oficiales (dicho de otra forma, obtiene un índice de actualizaciones)
upgrade: opción de apt-get que actualiza el sistema.
install: es la opción de apt-get que indica que se instalarán uno o más paquetes

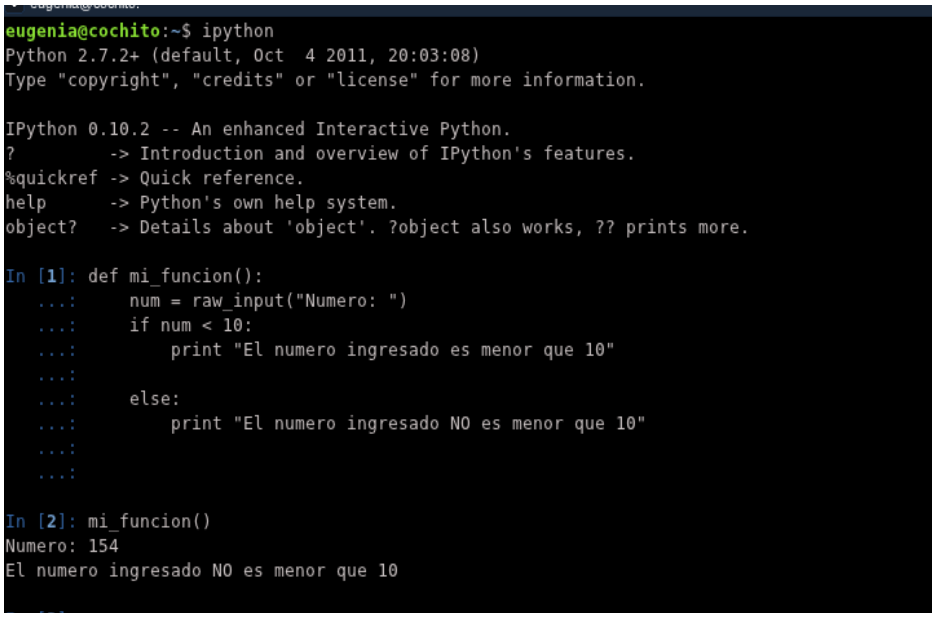
Instalación de un Shell interactivo mejorado

Python trae por defecto su propio Shell interactivo, el cuál nos permite escribir código Python y ejecutarlo. Sin embargo, tenemos la opción de contar con un Shell interactivo mejorado, que entre otras ventajas sobre el shell nativo, podemos encontrar números de línea, sangrado automático, etc.

iPython, es el Shell interactivo que elegiremos. Para instalarlo, ejecuta la siguiente orden desde una terminal:

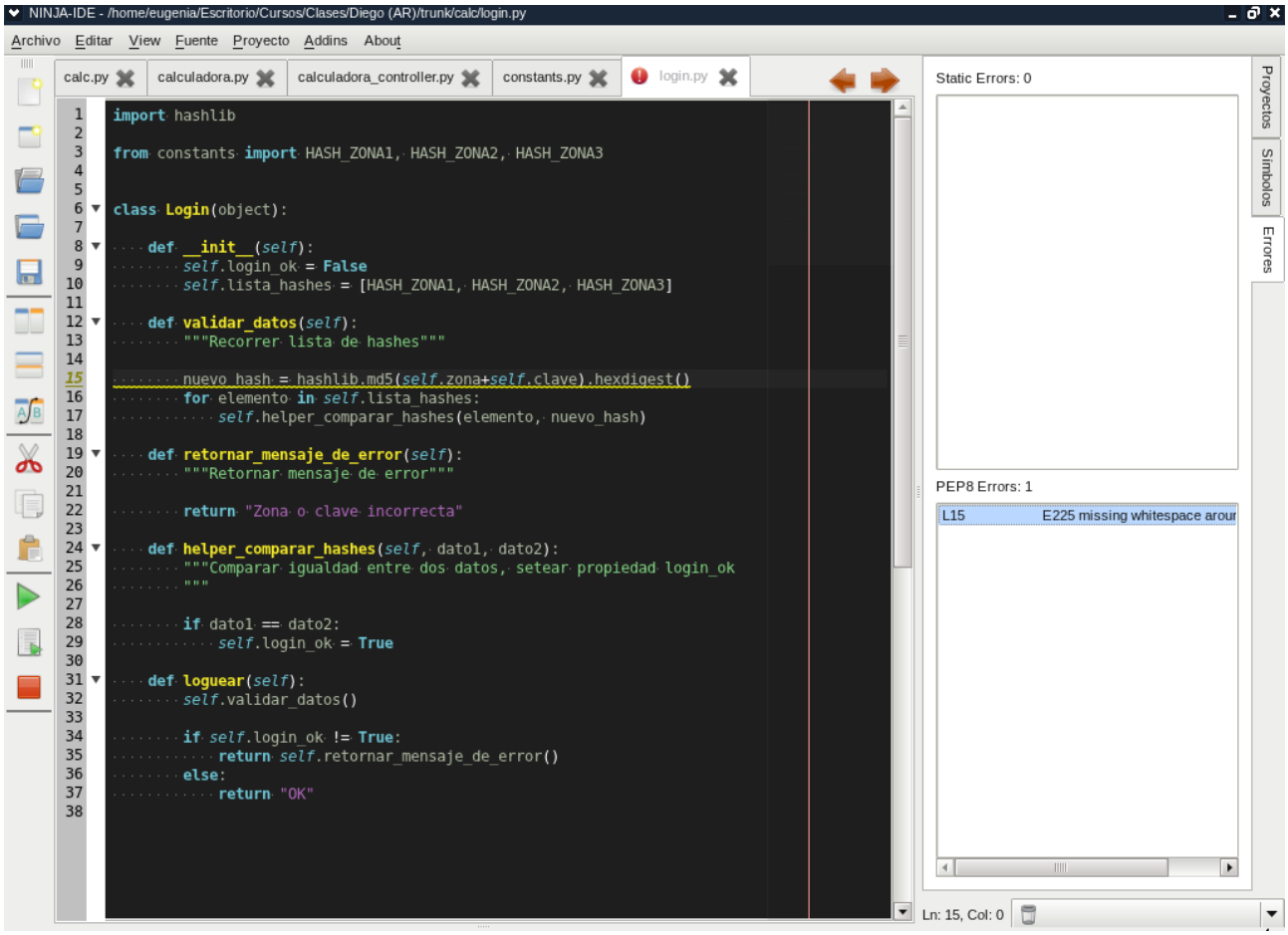
```
eugenia@cochito:~$ sudo apt-get install ipython
```

Para ejecutar el nuevo shell interactivo, solo deberás escribir el comando ipython:



Ninja-IDE

Ninja-IDE es un Entorno Integrado de Desarrollo[1] que nos permitirá, crear proyectos en Python, al tiempo de ir ejecutando nuestros códigos y corrigiendo eventuales errores que éstos, puedan presentar.



Para instalar Ninja-IDE en tu ordenador, desde la terminal, ejecuta los siguientes comandos:

1) Agrega el PPA de Ninja-IDE:

```
sudo apt-add-repository ppa:ninja-ide-developers/daily
```

2) Sincroniza el índice de actualizaciones:

```
sudo apt-get update
```

3) Instala Ninja-IDE:

```
sudo apt-get install ninja-ide
```

1 http://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado

Instalación de Bazaar

Bazaar es un sistema de control de versiones distribuido, que nos permitirá ir manteniendo el control cambios sobre nuestros archivos, centralizándolos en un repositorio.

Un Repositorio es un espacio destinado a almacenar información digital. En nuestro caso, lo que se almacenará en ese repositorio, serán los archivos -código fuente, *tarballs*, binarios, etc- de las aplicaciones y ejercicios que iremos *codeando* a lo largo del curso.

Las ventajas principales de utilizar un SCV, son:

- Espacio de **almacenamiento centralizado** de, principalmente, el código fuente de la aplicación así como scripts de construcción -en el caso de aplicaciones que requieran ser compiladas o simplemente, necesiten realizar configuraciones especiales, ya sea tanto para continuar desarrollándolas como para ejecutarlas-.
- Para ser efectivos, deben llevar un **control histórico de cambios** que se vayan efectuando en los archivos -preferentemente automático-, permitir el establecimiento de tags -etiquetas- que ayuden a identificar diferentes releases -versiones-.

Los Sistemas de Control de Versiones (SCV) pueden agruparse en dos tipos:

- **Centralizados:**

  un único repositorio centralizado administrado por un solo responsable.
- **Distribuidos (recomendados):**

  donde existe un repositorio central que cada usuario podrá clonar para obtener su propio repositorio -local- e interactuar con con otros repositorios locales.

Entre los **SCV distribuidos** podemos destacar excelentes alternativas **GPL** (Software Libre), como es el caso de -entre otros-, **Git** (de Linus Torvalds, creador del Kernel Linux en el que se basa el Sistema Operativo GNU/Linux), **Mercurial** (desarrollado en Python y C) o el magnífico **Bazaar**, nacido a partir de GNUArch y desarrollado íntegramente en Python por Martin Pool, con el patrocinio de Canonical y **elegido en este curso**.

> Una gran **ventaja de los SCV** es que **permiten a varios programadores trabajar simultáneamente sobre los mismos archivos**, impidiendo que el trabajo de uno, pise al trabajo de otro.

Los SCV pueden utilizarse tanto a través de línea de comandos, como de aplicaciones

gráficas. En este curso, nos centraremos en el uso por medio de línea de comandos.

Los SCV, en su mayoría -y a rasgos generales- cuentan con un conjunto de funcionalidades, las cuales, para cada una, existe un determinado comando (generalmente, similar en la mayoría de los SCV).

Para **instalar Bazaar en tu ordenador**, ejecuta el siguiente comando:

```
sudo apt-get install bzr
```

Una vez instalado Bazaar, deberás **clonar el repositorio central** (desde el servidor del curso) a tu ordenador local:

```
bzr branch sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
```

A continuación, deberás ingresar tu contraseña.

Una vez clonado el repositorio, deberás agregar unas líneas al archivo de configuración de Bazaar. Para ello, abre el archivo de configuración con el editor Nano:

```
nano trunk/.bzr/branch/branch.conf
```

Mueve el cursor hasta la siguiente línea y pulsa las teclas Ctrl + K:

```
parent_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
```

A continuación, pulsa tres veces, las teclas Ctrl + U para pegar (tres veces) la línea que cortaste anteriormente. Deberás ver lo siguiente:

```
parent_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
parent_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
parent_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
```

Reemplaza la palabra "*parent*" de la segunda línea, por "*push*" y la de la tercera, por "*pull*" de forma tal que el archivo, se vea como sigue:

```
parent_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
push_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
pull_location = sftp://tu_usuario@66.228.52.93/home/tu_usuario/public/trunk
```

Para guardar el archivo pulsa las teclas Ctrl + O (enter) y para salir, pulsa Ctrl + X.

02 Estructura y elementos del lenguaje

Dentro de los **lenguajes informáticos**, Python, pertenece al grupo de los **lenguajes de programación** y puede ser clasificado como un **lenguaje interpretado**, **de alto nivel**, **multiplataforma**, de **tipado dinámico** y **multiparadigma**. A diferencia de la mayoría de los lenguajes de programación, **Python nos provee de reglas de estilos**, a fin de poder escribir código fuente más legible y de manera estandarizada. Estas reglas de estilo, son definidas a través de la ***Python Enhancement Proposal* Nº 8 (PEP 8)** , la cual iremos viendo a lo largo del curso.

GLOSARIO

Lenguaje informático:es un idioma artificial, utilizado por ordenadores, cuyo fin es transmitir información de algo a alguien. Los lenguajes informáticos, pueden clasificarse en: a) lenguajes de programación (Python, PHP, Pearl, C, etc.); b) lenguajes de especificación (UML); c) lenguajes de consulta (SQL); d) lenguajes de marcas (HTML, XML); e) lenguajes de transformación (XSLT); f) protocolos de comunicaciones (HTTP, FTP); entre otros.

Lenguaje de programación: es un lenguaje informático, diseñado para expresar órdenes e instrucciones precisas, que deben ser llevadas a cabo por una computadora. El mismo puede utilizarse para crear programas que controlen el comportamiento físico o lógico de un ordenador. Está compuesto por una serie de símbolos, reglas sintácticas y semánticas que definen la estructura del lenguaje.

Lenguajes de alto nivel: son aquellos cuya característica principal, consiste en una estructura sintáctica y semántica legible, acorde a las capacidades cognitivas humanas. A diferencia de los lenguajes de bajo nivel, son independientes de la arquitectura del hardware, motivo por el cual, asumen mayor portabilidad.

Lenguajes interpretados: a diferencia de los compilados, no requieren de un compilador para ser ejecutados sino de un intérprete. Un intérprete, actúa de manera casi idéntica a un compilador, con la salvedad de que ejecuta el programa directamente, sin necesidad de generar previamente un ejecutable. Ejemplo de lenguajes de programación interpretado son Python, PHP, Ruby, Lisp, entre otros.

Tipado dinámico: un lenguaje de tipado dinámico es aquel cuyas variables, no requieren ser definidas asignando su tipo de datos, sino que éste, se auto-asigna en tiempo de ejecución, según el valor declarado.

Multiplataforma: significa que puede ser interpretado en diversos Sistemas Operativos como GNU/Linux, Windows, Mac OS, Solaris, entre otros.

Multiparadigma: acepta diferentes paradigmas (técnicas) de programación, tales como la orientación a objetos, aspectos, la programación imperativa y funcional.

Código fuente: es un conjunto de instrucciones y órdenes lógicas, compuestos de algoritmos que se encuentran escritos en un determinado lenguaje de programación, las cuales deben ser interpretadas o compiladas, para permitir la ejecución del programa informático.

• • •

Hasta aquí la previsualización.