

# Vim + Python-mode



Carlos Gustavo Ruiz  
@atmantree

# Agenda

- IDEs vs Editores
- Vi → Vim
- Vim
- Personalización,  
Scripts y Plugins
- Python-Mode
- Recursos

# IDEs vs Editores de Texto

## IDEs

- Integrado (todo incluido)
- Alto consumo de recursos
- Uso de Entorno Gráfico es indispensable
- Generalmente definen un estilo de hacer las cosas.

## Editores de Texto

- Extensibles
- Bajo consumo de recursos
- Uso de Entorno Gráfico es opcional
- Normalmente no interviene en el estilo del hacer las cosas.

# IDEs vs Editores de Texto

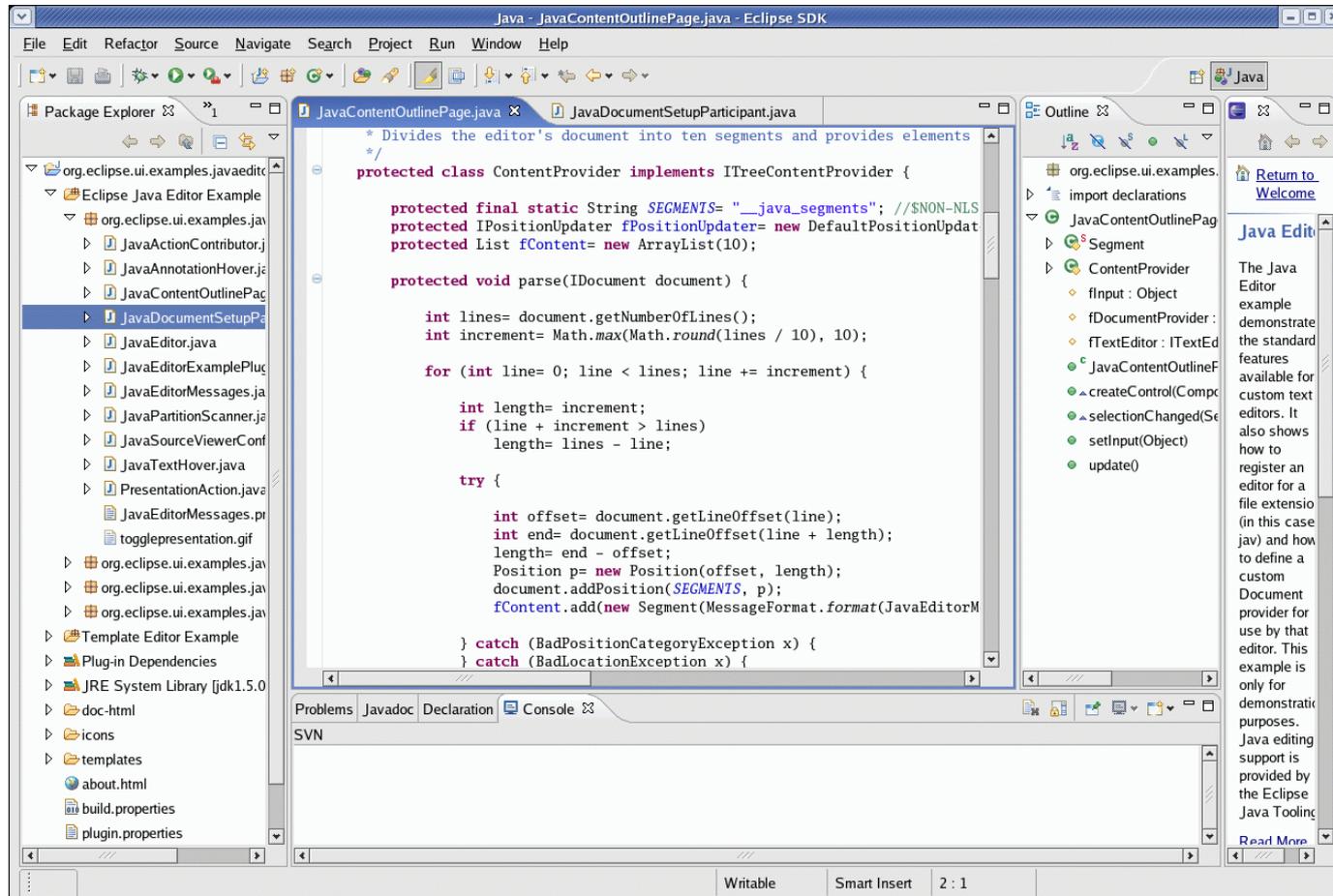
## IDEs

- Eclipse
- NetBeans
- VisualStudio
- Xcode
- Qt Creator
- Anjuta

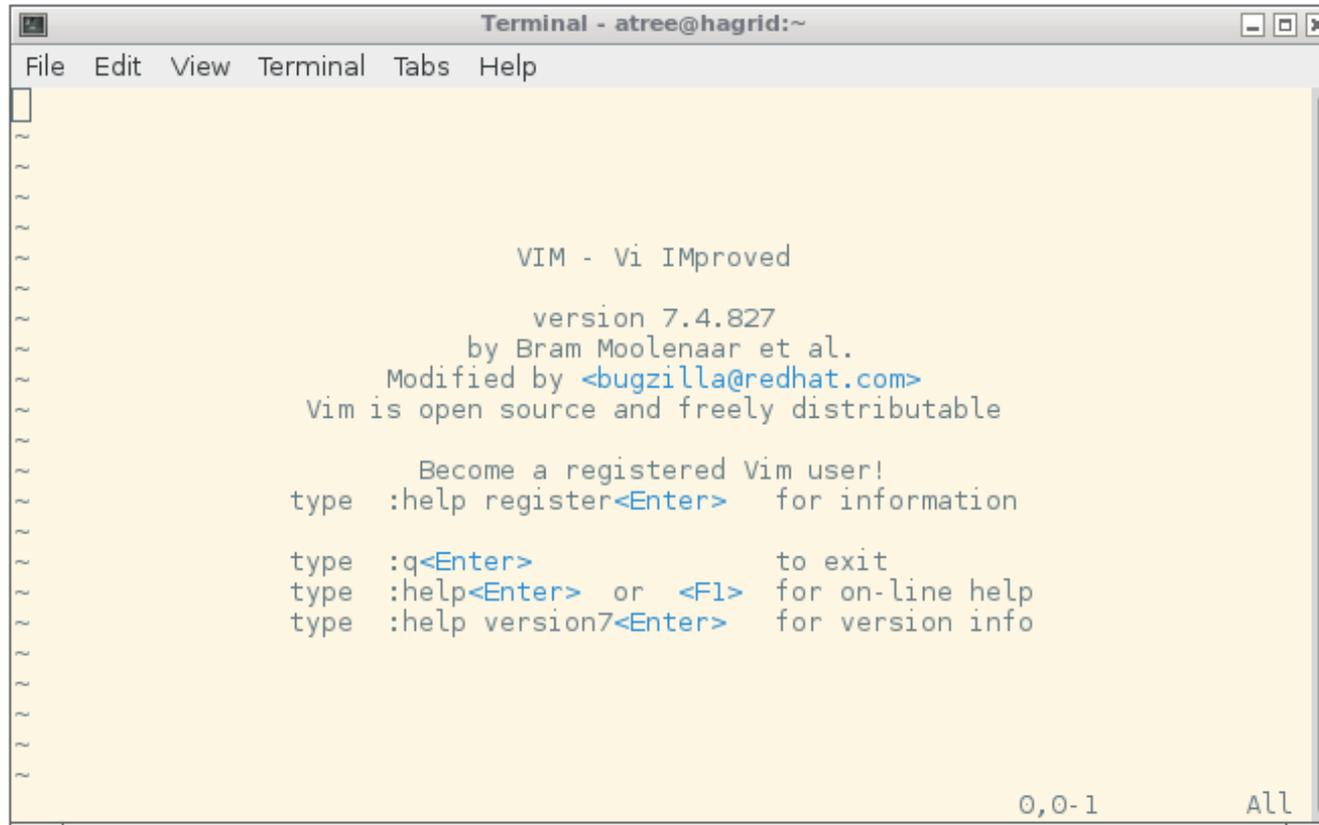
## Editores de Texto

- Vim
- Emacs
- Notepad++
- SublimeText
- Atom
- nano

# IDEs vs Editores de Texto



# IDEs vs Editores de Texto



The image shows a terminal window titled "Terminal - atree@hagrid:~". The window contains the following text:

```
File Edit View Terminal Tabs Help

VIM - Vi IMproved

version 7.4.827
  by Bram Moolenaar et al.
  Modified by <bugzilla@redhat.com>
  Vim is open source and freely distributable

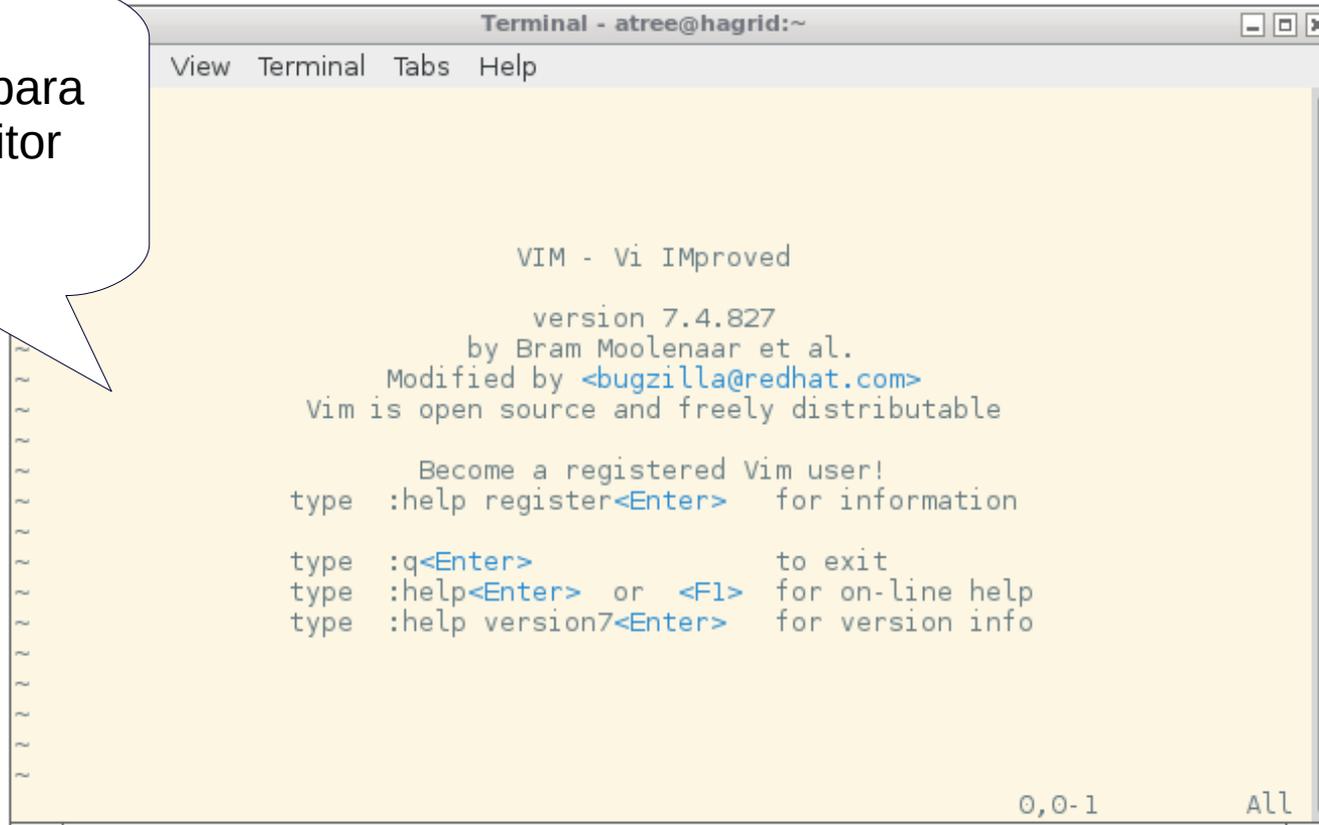
  Become a registered Vim user!
type  :help register<Enter>  for information

type  :q<Enter>              to exit
type  :help<Enter> or <F1>   for on-line help
type  :help version7<Enter>  for version info

0,0-1 All
```

# IDEs vs Editores de Texto

¿Una charla para un simple editor de texto?

A terminal window titled "Terminal - atree@hagrid:~" with a menu bar containing "View", "Terminal", "Tabs", and "Help". The terminal content displays the Vim startup screen, including the version number 7.4.827 and instructions for help and exiting. The status bar at the bottom right shows "0,0-1" and "All".

```
Terminal - atree@hagrid:~
View Terminal Tabs Help

VIM - Vi IMproved

version 7.4.827
by Bram Moolenaar et al.
Modified by <bugzilla@redhat.com>
Vim is open source and freely distributable

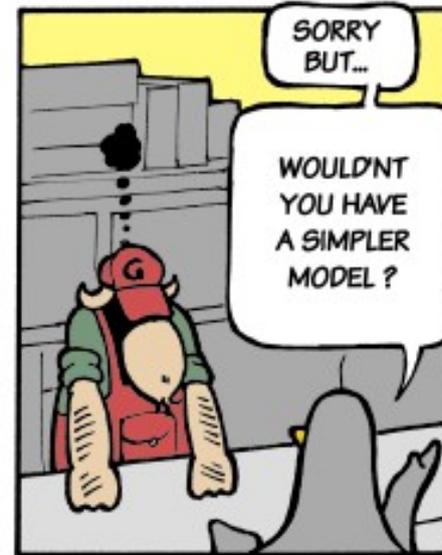
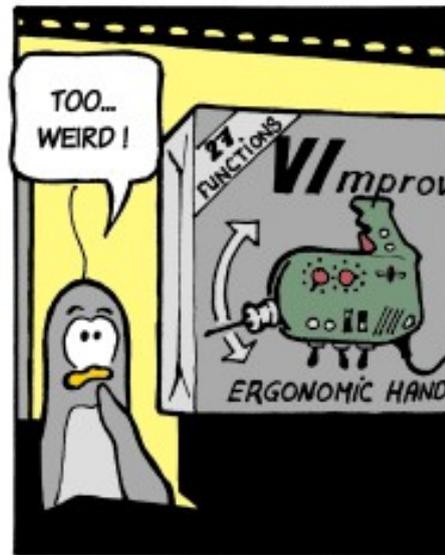
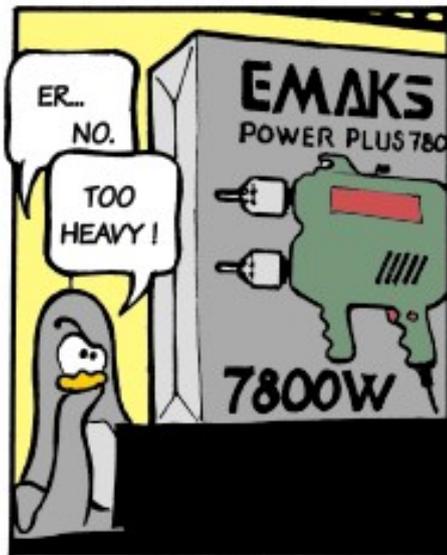
Become a registered Vim user!
type :help register<Enter> for information

type :q<Enter> to exit
type :help<Enter> or <F1> for on-line help
type :help version7<Enter> for version info

0,0-1 All
```



# IDEs vs Editores de Texto



# ¿Por qué Vim?

- Vim no solo es un editor, es una herramienta.
- Es rápido.
- Muy flexible y altamente personalizable.
- Se ejecuta prácticamente en cualquier plataforma.
- Trabaja con muchísimos lenguajes de programación.
- Extensible (Macros, Scripts y Plugins).
- Es Código Abierto (Charityware)

# Vi → Vim

- **vi** → 1976 (modo visual para editor ex, incluye el concepto de edición modal)
- **STEVIE** → 1987 (primer clon de vi, ofrecía un grupo limitado de opciones de vi, diseñado para Atari ST, luego portado a OS/2) *10% compatible*
- **Elvis** → 1997 (corregía errores de STEVIE, incluyo resaltado de sintaxis, multiples ventanas y una gui, era el vi estándar de Slackware y Minix en el 2003) *80% compatible*
- **nvi** → 2001 (Derivado de Elvis, multiples buffers, historial ilimitado, expresiones regulares, scripting con Perl y Tcl/tk, distribuido con los Unix BSD) *95% compatible*
- **Vim** → 1991 (Edición de multiples archivos en multiples buffers, ventanas y pestañas, lenguaje de scripting, soporte de Perl y Python, resaltado de código de más de 200 lenguajes de programación, historial ilimitado, expresiones regulares, completado de palabras de acuerdo al contexto, integración con multiples compiladores) *99% compatible*
- **Vile** → 1990 (Intento por crear un editor con lo mejor de vi y emacs, edición modal, lenguaje procedimental, soporte para perl – experimental, funciones pueden mapearse a teclas) *10% compatible*

# Vim

- Edición Modal
  - Modo Normal (desplazamiento y manipulación de texto)
  - Modo Insert (modifica el texto)
  - Modo Visual (resalta una porción del archivo y la modifica en bloque)
  - Modo Comandos (ejecuta acciones en forma de comandos)
  - Modo Ex (comandos)

# Vim

## Modo normal

- Desplazamiento (h, j, k, l) (izquierda, arriba, abajo, derecha)

# Vim

## Modo normal

- Desplazamiento (h, j, k, l) (izquierda, abajo, arriba, derecha)



<https://en.wikipedia.org/wiki/ADM-3A> (un estándar de 1974)  
[http://xahlee.info/kbd/keyboard\\_hardware\\_and\\_key\\_choices.html](http://xahlee.info/kbd/keyboard_hardware_and_key_choices.html)



# Vim

## Modo normal

- Otros desplazamientos (algunos de los más comunes)
  - `w` → Avanzar palabra
  - `W` → Avanzar hasta el próximo carácter en blanco
  - `b` → Retroceder al inicio de la palabra
  - `B` → Retroceder hasta el carácter en blanco previo
  - `e` → Avanzar hasta el final de palabra
  - `^` → Inicio de línea
  - `$` → Final de línea
  - `/texto` → ir a la vez que aparezca *texto*
  - `J` → Quitar salto de línea

# Vim

## Modo normal

- Otros desplazamientos (algunos de los más comunes)
  - ^E → Mover la ventana hacia abajo
  - ^Y → Mover la ventana hacia arriba
  - ^F → Bajar una página
  - ^B → Subir una página
  - H → Mover el cursor al tope de la ventana
  - M → Mover el cursor al medio de la ventana
  - L → Mover el cursor al fondo de la ventana
  - gg → Ir al inicio del archivo
  - G → Ir al final del archivo

vim - movement commands

back

forward

absolute movements

'' last location

'.' last edit

#G line #

% matching bracket

0 line

^ non-blank

Fx find x

Tx after x

B delimited word

b word

gE delimited end

ge end

h left

) sentence

C-d 1/2 page

previous

gg first line

# find word under cursor

N previous text

?text find text

C-b page

C-u 1/2 page

H screen

{ paragraph

( sentence

k up

j down

l right

e end

E delimited end

w word

W delimited word

tx before x

fx find x

;

next x

\$ line

/text find text

n next text

\* find word under cursor

G last line next



# Vim

## Objetos de texto

- w → palabras
- s → sentencias (oraciones)
- p → párrafos
- t → tags (etiquetas – xml/html)

## Movimientos

- a → all (todos)
- i → in (dentro)
- t → until (hasta)
- f → buscar hacia adelante
- F → buscar en reversa

## Comandos

- d → delete (borrar y/o cortar)
- c → change (borrar y cambiar a modo insert)
- y → yank (copiar)
- v → selección visual



# Vim

Ahora juntemos lo anterior con esta formula

- **[numero]{comando}{objeto o movimiento}**

Ejemplos de acciones:

- diw → delete in word (borrar en palabra)
- ciw → change in word (igual que el anterior pero te cambia a modo insert)
- caw → cambiar toda la palabra
- yi) → copiar el contenido dentro de paréntesis
- di) → borrar el contenido dentro del paréntesis
- da) → igual que el anterior pero elimina el paréntesis inclusive

# Vim

- Otros comandos:
  - dd → borrar línea
  - yy → copiar línea
  - p → pegar en la línea siguiente
  - P → pegar en la línea previa
  - D → borrar hasta el final de la línea
  - C → cambiar hasta el final de la línea
  - o → insertar línea después de la actual
  - O → insertar línea después de la actual
  - I → mover al inicio y pasar a modo edición
  - A → mover al final y pasar a modo edición
- Para que averiguar:
  - ¿De lo visto que hace va”?
  - ¿Qué hace el comando . (punto)?

# Vim

## Modo Insert

- Se usa para modificar texto como haría con un editor común y corriente

## Modo visual

- Permite seleccionar un bloque y aplicar comandos y modificaciones por lote

## Modo Comando

- Es todo lo que escribimos en los comandos con la fórmula :comando.
- Ejemplo:
  - :help
  - :vimtutor

## Modo ex

- Parecido al modo comando, permite ejecutar instrucciones del comando ex:

```
vim -E -s Makefile <<-EOF
:%substitute/CFLAGS = -g$/CFLAGS =-fPIC -DPIC -g/
:%substitute/CFLAGS =$/CFLAGS =-fPIC -DPIC/
:%substitute/ADAFLAGS =$/ADAFLAGS =-fPIC -DPIC/
:update
:quit
EOF
```

# Personalización, Scripts y Plugins

- Una vez que se ha personalizado Vim, este queda como su chaqueta o el jean favorito.
  - Para personalizar Vim principalmente nos valemos de modificar el archivo *.vimrc*
  - En el se hace mapeado de atajos de teclado, cambiamos de esquema de colores, se corre rutinas y se define condiciones y sintaxis para diferentes tipos de archivos y un montón de cosas más.

# Personalización, Scripts y Plugins

Vim es extensible vía Macros, Scripts y Plugins

- Las macros permiten simplificar rutinas simples durante la edición
- Los scripts en VimL permiten automatizar tareas más complejas y eventualmente escribir..
- Plugins, que es de lo más genial una vez que se comprenden los conceptos básicos.

# Personalización, Scripts y Plugins

## Plugins

- Instalación:
  - Instalarlos a mano en la carpeta .vim/ con algo de trabajo sobre el archivo .vimrc (ejemplo <http://stackoverflow.com/a/1639654>)
  - Usar gestor de plugins:
    - Vundle
    - Pathogen
    - Neobundle Dein



# Plugins Útiles

- Syntastic

The screenshot shows a Gvim editor window titled "main.cpp (~/projects/hansolo/src) - GVIM". The code is as follows:

```
#include "engine.h"
#include <iostream>

int main(int argc, char argv[]) {
    if (argc != 2) {
        std::cout << "Usage: " << argv[0] << " e.g. " << argv[0] << " ../"
        exit(0);
    }
    string map_path(argv[1]);
    if (*map_path.end() != '/')
        map_path.append("/");

    Engine engine(map_path);
    try {
        engine.main_loop();
    } catch(exception* e) {
        engine.teardown_curses();
        cout << "Exception caught: " << e->what() << endl;
    } catch(exception e) {
```

Annotations in the image point to various Syntastic features:

- 1. Location list**: Points to the bottom statusline showing the location list: `[Location List]` and `invalid conversion from 'char' to 'const char*' [-fpermissive]`.
- 2. Command window**: Points to the command window showing the error: `main.cpp|10 col 28 error| invalid conversion from 'char' to 'const char*' [-fpermissive]`.
- 3. Signs**: Points to the sign column on the left side of the editor.
- 4. Statusline flag**: Points to the statusline showing the error: `[Syntax: line:4 (3)]`.
- 5. Error balloons**: Points to the error balloon showing the message: `second argument of 'int main(int, char*)' should be 'char **' [-Wmain]`.

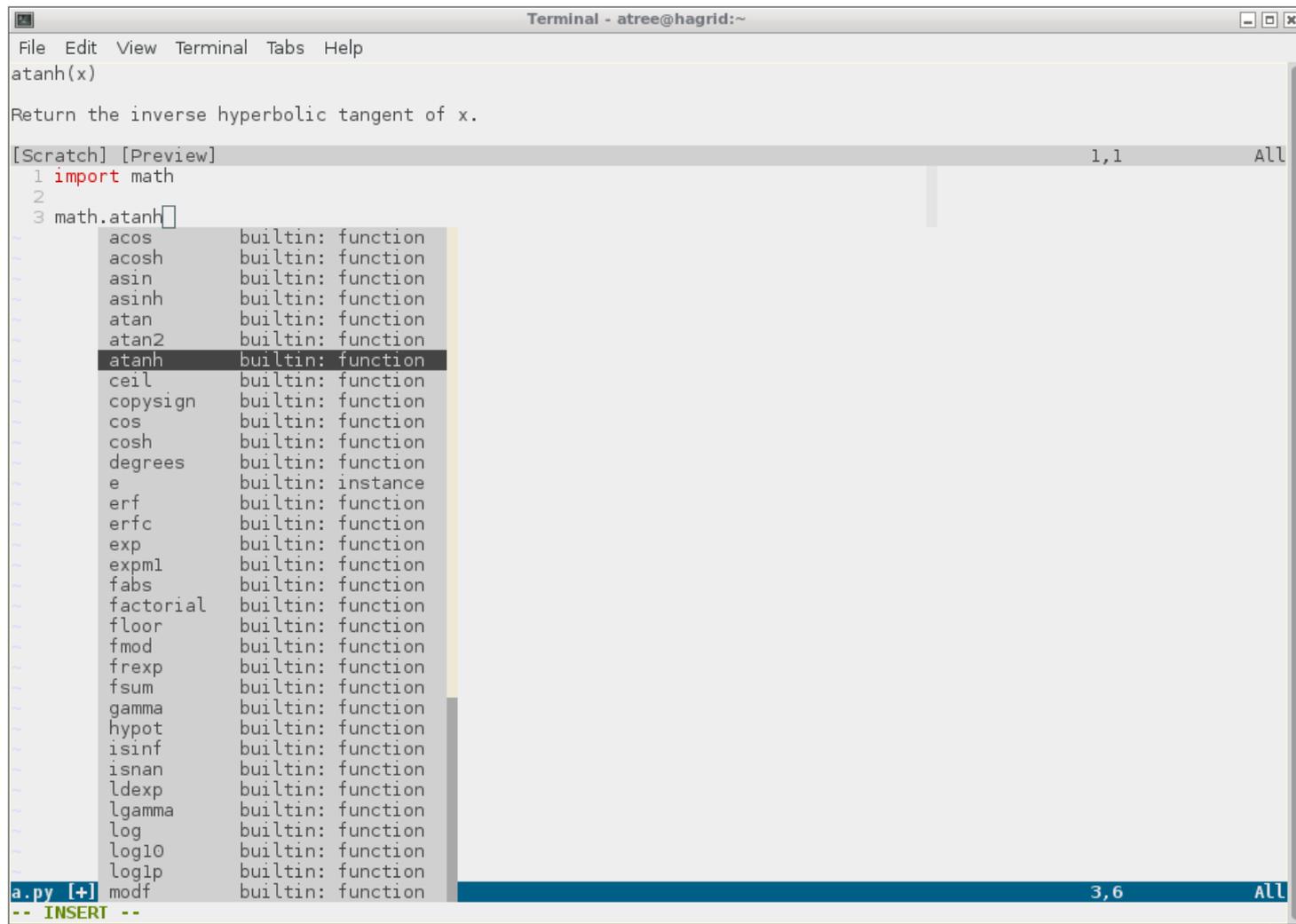
# Plugins Útiles

- Figitive

```
Terminal - atree@hagrid:~/Workspace/apvyve
File Edit View Terminal Tabs Help
70480470 (J. Hernan Ramirez R 2015-07-30 20:20:35 -0430) ## Recursos compartidos vía la comunidad de Python Venezuel
70480470 (J. Hernan Ramirez R 2015-07-30 20:20:35 -0430)
18c497bf (satanas 2015-08-07 18:23:32 -0300) - Índice:
5e1e4201 (J. Hernan Ramirez R 2015-08-09 12:17:44 -0430)   - [Análisis científico con Python](#análisis-científico
6bb69f48 (J. Hernan Ramirez R 2015-08-09 13:15:53 -0430)   - [Frameworks basados en Python](#frameworks-basados-en
c559af59 (J. Hernan Ramirez R 2015-08-08 06:36:27 -0430)   - [Hardware libre con Python](#hardware-libre-con-pytho
cab3c329 (J. Hernan Ramirez R 2015-08-09 18:40:41 -0430)   - [Herramientas para desarrollo y depuración de código]
4194d9a8 (Carlos Gustavo Ruiz 2015-08-12 22:55:30 -0430)   - [Herramientas para gestión de proyectos](#herramienta
4194d9a8 (Carlos Gustavo Ruiz 2015-08-12 22:55:30 -0430)   - [Recursos para el desarrollo de backends](#recursos-p
4194d9a8 (Carlos Gustavo Ruiz 2015-08-12 22:55:30 -0430)   - [Recursos para el manejo de bases de datos](#recursos
4194d9a8 (Carlos Gustavo Ruiz 2015-08-12 22:55:30 -0430)   - [Recursos para el desarrollo de front-ends](#recursos
18c497bf (satanas 2015-08-07 18:23:32 -0300)   - [Recursos para el despliegue de aplicaciones](#recurs
0900cb38 (satanas 2015-08-09 22:41:04 -0300)   - [Servicios para el control de versiones](#servicios-p
34eb9795 (satanas 2015-08-09 22:00:11 -0300)   - [Recursos para crear interfaces gráficas](#recursos-p
56909cd9 (Francisco Palm 2015-08-10 12:23:18 -0430)   - [Otros recursos útiles](#otros-recursos-útiles)
18c497bf (satanas 2015-08-07 18:23:32 -0300)   - [Blog posts](#blog-posts)
70480470 (J. Hernan Ramirez R 2015-07-30 20:20:35 -0430)   - - -
70480470 (J. Hernan Ramirez R 2015-07-30 20:20:35 -0430)
70480470 (J. Hernan Ramirez R 2015-07-30 20:20:35 -0430)
5e1e4201 (J. Hernan Ramirez R 2015-08-09 12:17:44 -0430) ## Análisis científico con Python
c43cec49 (J. Hernan Ramirez R 2015-07-30 20:52:09 -0430)
56909cd9 (Francisco Palm 2015-08-10 12:23:18 -0430) ### Paquetes básicos
56909cd9 (Francisco Palm 2015-08-10 12:23:18 -0430)
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) * [Numpy](http://www.numpy.org/) - es el paquete fundamenta
8b4913fc (Carlos Gustavo Ruiz 2015-08-13 11:33:04 -0430) cómputo numérico. Permite definir arreglos y matrices numér
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) multidimensionales y realizar operaciones sobre estos.
481107b4 (Carlos Gustavo Ruiz 2015-08-12 22:27:42 -0430)
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) * [Scipy](http://www.scipy.org/) - La biblioteca SciPy es u
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) algortimos numéricos y cajas de herramientas para dominios
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) incluyen procesamiento de señales, optimización, estadístic
481107b4 (Carlos Gustavo Ruiz 2015-08-12 22:27:42 -0430)
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) * [Pandas](http://pandas.pydata.org/) - Biblioteca de alto
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) de usar para realizar análisis y modelado de datos *(en eng
481107b4 (Carlos Gustavo Ruiz 2015-08-12 22:27:42 -0430)
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) * [SymPy](http://www.sympy.org/) - es una biblioteca para m
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) Tiene el objetivo de convertirse en un sistema completo de
e65fb756 (Carlos Gustavo Ruiz 2015-08-12 22:48:42 -0430) implementado totalmente en Python.
56909cd9 (Francisco Palm 2015-08-10 12:23:18 -0430)
56909cd9 (Francisco Palm 2015-08-10 12:23:18 -0430) ### Gráficos
56909cd9 (Francisco Palm 2015-08-10 12:23:18 -0430)
/tmp/vmfmOPK/1.fugitiveblame 31,1 Top README.md 31,0-1 Top
```

# Plugins Útiles

- YouCompleteMe



```
Terminal - atree@hagrid:~
File Edit View Terminal Tabs Help
atanh(x)
Return the inverse hyperbolic tangent of x.
[Scratch] [Preview] 1,1 All
1 import math
2
3 math.atanh(
  acos      builtin: function
  acosh     builtin: function
  asin      builtin: function
  asinh     builtin: function
  atan      builtin: function
  atan2     builtin: function
  atanh     builtin: function
  ceil      builtin: function
  copysign  builtin: function
  cos       builtin: function
  cosh     builtin: function
  degrees   builtin: function
  e         builtin: instance
  erf       builtin: function
  erfc     builtin: function
  exp       builtin: function
  expm1    builtin: function
  fabs     builtin: function
  factorial builtin: function
  floor    builtin: function
  fmod     builtin: function
  frexp    builtin: function
  fsum     builtin: function
  gamma    builtin: function
  hypot    builtin: function
  isinf    builtin: function
  isnan    builtin: function
  ldexp    builtin: function
  lgamma   builtin: function
  log      builtin: function
  log10    builtin: function
  loglp    builtin: function
  modf     builtin: function
a.py [+]
```

# Python Mode

Python Mode es un plugin para facilitar el trabajo con Python dentro de VIM, incluye un grupo de herramientas entre las que se cuentan:

- Pylint → Analizador de código fuente
- Rope → Herramienta de refactorización. Ayuda a buscar las definiciones de objetos y funciones.
- PyDoc → Herramienta para obtener la documentación de los módulos utilizados.
- PyFlakes → Validaciones dentro del código.
- Pep8 → Para estándares de codificación.
- Pep257 → Alerta de Docstrings faltantes.
- mccabe → Análisis de complejidad ciclomática

# Python Mode

Entre las bondades de python-mode se cuentan:

- Soporte para Python 2 y Python 3
- Resaltado de sintaxis mejorado
- Soporte para virtualenv
- Ejecutar código python
- Agregar y quitar puntos de parada (breakpoints)
- Indentación mejorada
- Folding (compactación)
- Movimientos definidos para Python
- Revisión de código
- Autoarreglo de errores de PEP 8
- Búsqueda en documentación de Python
- Ir a definición
- Y más..

# Python Mode

Demo

# Recursos

- <http://www.vim.org/>
- Para aprender
  - vimtutor
  - <http://www.openvim.com/>
  - <http://vim-adventures.com/>
  - <http://www.vimgenius.com/>
- Para Plugins
  - <http://vimawesome.com/>

# Recursos

- Para consultas rápidas
  - :help
  - <http://vim.wikia.com/>
- Configuraciones “llave en mano”
  - <http://vim-bootstrap.com/>
  - <http://vim.spf13.com/>
- Comunidad Vim de Venezuela
  - <https://telegram.me/vimvnzla>

# Referencias

- [https://en.wikibooks.org/wiki/Learning\\_the\\_vi\\_Editor/Vim](https://en.wikibooks.org/wiki/Learning_the_vi_Editor/Vim)
- <https://youtu.be/5r6yzFEXajQ>
- <https://youtu.be/YhqsjUUHj6g>
- <http://www.vim.org/6k/features.es.txt>
- [http://vim.wikia.com/wiki/Vim\\_Tips\\_Wiki](http://vim.wikia.com/wiki/Vim_Tips_Wiki)