Python简介与环境搭建

# Python 简介

Python 是一个高层次的结合了解释性、编译性、互动性和面向对象的脚本语言。Python 的设计具有很强的可读性，相比其他语言经常使用英文关键字，其他语言的一些标点符号，它具有比其他语言更有特色语法结构。

* Python 是一种解释型语言： 这意味着开发过程中没有了编译这个环节。类似于PHP和Perl语言。
* Python 是交互式语言： 这意味着，您可以在一个Python提示符，直接互动执行写你的程序。
* Python 是面向对象语言: 这意味着Python支持面向对象的风格或代码封装在对象的编程技术。
* Python 是初学者的语言：Python 对初级程序员而言，是一种伟大的语言，它支持广泛的应用程序开发，从简单的文字处理到 WWW 浏览器再到游戏。

## Python 发展历史

Python 是由 Guido van Rossum 在八十年代末和九十年代初，在荷兰国家数学和计算机科学研究所设计出来的。Python 本身也是由诸多其他语言发展而来的,这包括 ABC、Modula-3、C、C++、Algol-68、SmallTalk、Unix shell 和其他的脚本语言等等。像 Perl 语言一样，Python 源代码同样遵循 GPL(GNU General Public License)协议。现在 Python 是由一个核心开发团队在维护，Guido van Rossum 仍然占据着至关重要的作用，指导其进展。

## Python 特点

|  |
| --- |
| **易于学习：**Python有相对较少的关键字，结构简单，和一个明确定义的语法，学习起来更加简单。  **易于阅读：**Python代码定义的更清晰。  **易于维护：**Python的成功在于它的源代码是相当容易维护的。  **一个广泛的标准库：**Python的最大的优势之一是丰富的库，跨平台的，在UNIX，Windows和Macintosh兼容很好。  **互动模式：**互动模式的支持，您可以从终端输入执行代码并获得结果的语言，互动的测试和调试代码片断。  **可移植：**基于其开放源代码的特性，Python已经被移植（也就是使其工作）到许多平台。  **可扩展：**如果你需要一段运行很快的关键代码，或者是想要编写一些不愿开放的算法，你可以使用C或C++完成那部分程序，然后从你的Python程序中调用。  **数据库：**Python提供所有主要的商业数据库的接口。  **GUI编程：**Python支持GUI可以创建和移植到许多系统调用。  **可嵌入:**你可以将Python嵌入到C/C++程序，让你的程序的用户获得"脚本化"的能力。 |

# Linux中安装python

## 下载安装python

地址：<https://www.python.org/downloads/release/python-2710/>

## CentOS安装Python的依赖包

yum install zlib-devel bzip2-devel openssl-devel ncurses-devel sqlite-devel readline-devel tk-devel gcc cmake -y

## 下载Python3.4的源码包并编译

|  |
| --- |
| wget <http://mirrors.sohu.com/python/3.4.1/Python-3.4.1.tar.xz>  xz -d Python-3.4.1.tar.xz  tar xf Python-3.4.1.tar -C /usr/local/src/  cd /usr/local/src/Python-3.4.1/  ./configure --prefix=/usr/local/python  make -j8 && make install |

## 添加命令到环境变量

|  |
| --- |
| vi /etc/profile  export PATH=$PATH:/usr/local/python/bin  source /etc/profile  ###测试  python3 –version |

# window中安装配置python环境

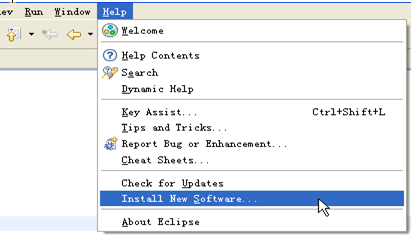
## 下载与安装

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

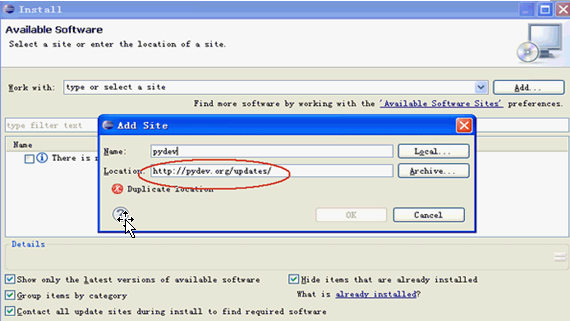
配置环境变量：D:\python

## 安装Pydev

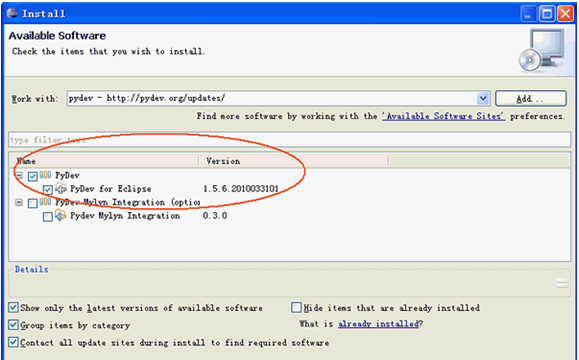
第一步：打开eclipse，help-->install new software



第二步：点击Add，添加pydev的安装地址：<http://pydev.org/updates/>

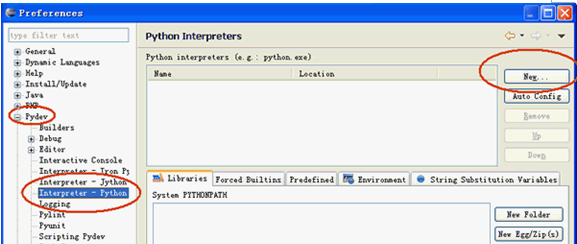


第三步：完成后点击"ok"，接着点击PyDev的"+"，展开PyDev的节点，要等一小段时间，让它从网上获取PyDev的相关套件，当完成后会多出PyDev的相关套件在子节点里，勾选它们然后按next进行安装。如下图所示。

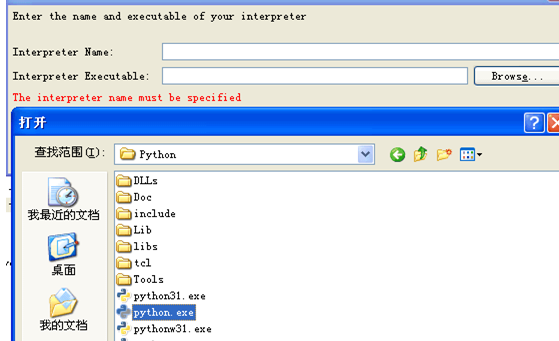


## 设置Pydev

安装完成后，还需要设置一下PyDev，选择Window -> Preferences来设置PyDev。设置Python的路径，从Pydev的Interpreter - Python页面选择New



会弹出一个窗口让你选择Python的安装位置，选择你安装Python的所在位置。



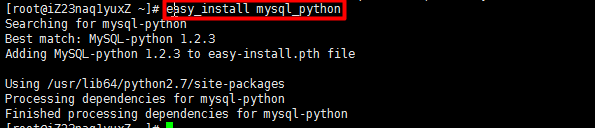
# Python依赖包的安装

## easy\_install的使用

windows下安装easy\_install可以<http://pypi.python.org/pypi/setuptools地方下载。并且把script>加入到环境变量中。

### 安装模块库

easy\_install PackageName



### 更新模块库

easy\_install -U PackageName

### 移除模块库

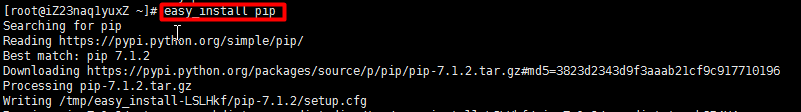
easy\_install -m PackageName

### 显示说明

easy\_install --help

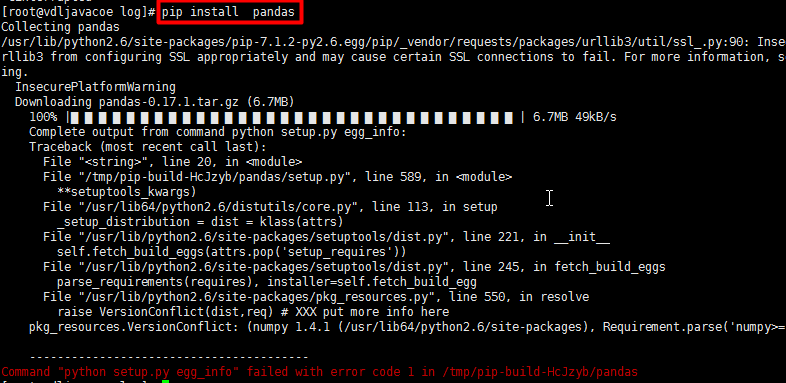
## pip的使用

#pip的安装：easy\_install pip



### 安装模块库

pip install PackageName



### 更新模块库

pip install -U PackageName

### 移除模块库

pip uninstall PackageName

### 搜索模块库

pip search PackageName

### 显示说明：

pip help

## 使用源码安装

|  |
| --- |
| $ gunzip MySQL-python-1.2.2.tar.gz  $ tar -xvf MySQL-python-1.2.2.tar  $ cd MySQL-python-1.2.2  $ python setup.py build  $ python setup.py install |
| --linux环境  yum install -y MySQL-python -y |

# Vim打造python编码 环境

运行下面两个命令，即可把 Vim（含插件）配置成 Python IDE。目前支持 MAC 和 Ubuntu。

|  |
| --- |
| curl -O https://raw.githubusercontent.com/vince67/v7\_config/master/vim.sh  bash vim.sh |

## vim基础配置(vimrc文件)

|  |
| --- |
| " 相关vim脚本文档 http://vimdoc.sourceforge.net  " 相关好文 https://github.com/yangyangwithgnu/use\_vim\_as\_ide  " 导入vim插件管理文件  if filereadable(expand("~/.vimrc.bundles"))  source ~/.vimrc.bundles  endif    " 基本配置  set nocompatible "不要vim模仿vi模式，建议设置，否则会有很多不兼容的问题  set mouse=a " 启动鼠标所有模式，但是右键功能不可用, 可以保证鼠标滚屏在当前屏幕内  set mousehide " 输入文件时隐藏鼠标  set backspace=indent,eol,start " 退格键分别可删除缩进，上一行结束，insert之前的字  set showmatch " 设置匹配模式  set nobackup " 不备份  set nowritebackup " 不写入备份文件  set noswapfile " 关闭交换文件  set history=500 " history存储长  set ruler " 显示标尺  set showcmd " 显示输入命令  set incsearch " 搜索时自动匹配  set hlsearch " 高亮搜索项  set ignorecase " 无视大小写  set smartcase " 如果有大写就区别大小写匹配  set laststatus=2 " 总是显示状态栏  " set autowrite " 切换文件自动保存 https://github.com/terryma/vim-multiple-cursors/raw/master/assets/example1.gif?raw=true  set shortmess=atI" " 关闭欢迎页面  set viewoptions=cursor,folds,slash,unix " viminfo 记录的内容  set virtualedit=onemore " 光标可以移到当行最后一个字符之后  set hidden " 切换文件不保存，隐藏  "set confirm " 退出前验证  " set spell " 拼写检查  set linespace=0 " 行之间没有多余的空格  set wildmenu " 自动补全时的文件菜单  set wildmode=list:longest,full " 自动补全时，匹配最长子串，列出文件  set whichwrap=b,s,h,l,<,>,[,] " 行尾可右移到下行，行首左移到上行,b：退格，s：空格，hl：左右，<>：n/v模式下的左右，[]：i/r模式下的左右  set scrolljump=5 " 光标离开屏幕范围  set scrolloff=3 " 光标移动至少保留行数    " 格式  set nowrap " 取消自动折行  "set smarttab  set tabstop=4 " tab=4空格  set softtabstop=4 " 回退可以删除缩进  set shiftwidth=4 " 缩进位宽=4个空格位  set autoindent " 自动缩进  set expandtab " tab由空格表示    set nojoinspaces " 用J合并两行用一个空格隔开  set splitright " 用vsplit新建窗口，让新的放右边  set splitbelow " 用split新建窗口，让新的放下面  set pastetoggle=<F12> " 指定F12进入黏贴模式，可以正常复制缩进  set iskeyword-=. " 让'.' 作为单词分割符  set iskeyword-=# " 让'#' 作为单词分割符  set iskeyword-=- " 让'-' 作为单词分割符  set listchars=tab:›\ ,trail:•,extends:#,nbsp:. " 空格等无效字符显示  set textwidth=80 " 内容宽度  set fileencodings=utf-8,gb18030,gbk,big5 " 文件编码    " 开启新的buffer时，自动转到对应文件目录  let g:autochdir = 1  if exists('g:autochdir')  autocmd BufEnter \* if bufname("") !~ "^\[A-Za-z0-9\]\*://" | lcd %:p:h | endif  endif    " 设置u的返回步数限制  if has('persistent\_undo')  set undofile " 开启u回滚文件记录  set undolevels=1000 " 最大数量的改变回滚  set undoreload=10000 " 最大数量重载可回滚行数  endif    " 特殊文件打开  autocmd BufNewFile,BufRead \*.py,\*.pyw set filetype=python  autocmd BufNewFile,BufRead \*.html.twig set filetype=html.twig  autocmd FileType haskell setlocal expandtab shiftwidth=2 softtabstop=2  autocmd BufNewFile,BufRead \*.coffee set filetype=coffee  autocmd FileType haskell setlocal commentstring=--\ %s  autocmd FileType haskell setlocal nospell    " key 映射  let mapleader = ',' " 全局leader设置  let maplocalleader = '\_' " 本地leader设置    " 设置tag和window间快速跳转  let g:easyWindows = 1  if exists('g:easyWindows')  " 向上  map <C-J> <C-W>j<C-W>\_  " 向下  map <C-K> <C-W>k<C-W>\_  " 向右  map <C-L> <C-W>l<C-W>\_  " 向左  map <C-H> <C-W>h<C-W>\_  endif    " 处理折叠行的左右移动  noremap j gj  noremap k gk    " 没sudo却想保存  cmap w!! w !sudo tee % >/dev/null    " ,fc查找冲突的地方  map <leader>fc /\v^[<\|=>]{7}( .\*\|$)<CR>    " ,ff 查找光标后的单词位置，列出选择项  nmap <Leader>ff [I:let nr = input("Which one: ")<Bar>exe "normal " . nr ."[\t"<CR>    " 屏幕左移和右移  map zl zL  map zh zH    " 映射vsp这些开启新的buffer,默认目录为当前目录  cnoremap %% <C-R>=fnameescape(expand('%:h')).'/'<cr>  map <leader>ew :e %%  map <leader>es :sp %%  map <leader>ev :vsp %%  map <leader>et :tabe %%    " 黏贴板  if has('clipboard')  if has('unnamedplus')  set clipboard=unnamedplus  else  set clipboard=unnamed  endif  endif    "字体的设置  set guifont=Consolas,Bitstream\_Vera\_Sans\_Mono:h9:cANSI "设置gui下的字体  set gfw=幼圆:h10:cGB2312    " 语法高亮  syntax on    " 缩进和md文件  filetype plugin indent on " 自动根据类型启动对应插件，缩进开启    " 设置主题,UI  " solarized 主题  set background=dark  if filereadable(expand("~/.vim/bundle/vim-colors-solarized/colors/solarized.vim"))  let g:solarized\_termcolors=256  let g:solarized\_termtrans=1  let g:solarized\_contrast="normal"  let g:solarized\_visibility="normal"  color solarized " Load a colorscheme  endif    set cursorline " 选中行高亮  autocmd BufEnter \*.py set cc=81 " 打开py文件81行高亮  " 之前的高亮线太难看，重新制定颜色，这里的black和iterm2颜色配置中的black一样  hi CursorLine cterm=NONE ctermbg=black guibg=black  hi CursorColumn cterm=NONE ctermbg=black guibg=black "  highlight ColorColumn ctermbg=black guibg=black  set number " 设置行号  hi LineNr ctermbg=black    " Indent Guides 缩进列对齐线  let g:indent\_guides\_start\_level = 2  let g:indent\_guides\_guide\_size = 1  let g:indent\_guides\_enable\_on\_vim\_startup = 1    " airline 就是状态栏的箭头  let g:airline\_theme="dark"  let g:airline\_powerline\_fonts = 1  let g:airline#extensions#tabline#enabled = 1  let g:airline#extensions#tabline#buffer\_nr\_show = 1    " NerdTree配置 control+e打开当前文件目录树  map <C-e> :NERDTreeToggle<CR>  map <leader>ee :NERDTreeToggle<CR>  nmap <leader>nt :NERDTreeFind<CR>    let NERDTreeShowBookmarks=1  let NERDTreeIgnore=['\.pyc', '\~$', '\.swo$', '\.swp$', '\.git', '\.hg', '\.svn', '\.bzr']  let NERDTreeChDirMode=0  let NERDTreeQuitOnOpen=1  let NERDTreeMouseMode=2  let NERDTreeShowHidden=1  let NERDTreeKeepTreeInNewTab=1  let g:nerdtree\_tabs\_open\_on\_gui\_startup=0    " syntastic 配置  " 设置每次w保存后语法检查  function! ToggleErrors()  Errors  endfunction  let g:syntastic\_check\_on\_open=1  let g:syntastic\_always\_populate\_loc\_list = 1  let g:syntastic\_auto\_loc\_list = 1  let g:syntastic\_check\_on\_wq = 0  let syntastic\_loc\_list\_height = 5  let g:syntastic\_html\_tidy\_ignore\_errors=[" proprietary attribute \"ng-"]  autocmd WinEnter \* if &buftype ==#'quickfix' && winnr('$') == 1 | quit |endif  autocmd WinLeave \* lclose    " 自动补全tab触发  function! InsertTabWrapper()  let col = col('.') - 1  if !col || getline('.')[col - 1] !~ '\k'  return "\<tab>"  else  return "\<c-p>"  endif  endfunction  inoremap <Tab> <c-r>=InsertTabWrapper()<cr>  inoremap <S-Tab> <c-n>    " Ctrlp 和 Ctrlp-funky(,fu)  if isdirectory(expand("~/.vim/bundle/ctrlp.vim/"))  let g:ctrlp\_working\_path\_mode = 'ra'  nnoremap <silent> <D-t> :CtrlP<CR>  nnoremap <silent> <D-r> :CtrlPMRU<CR>  let g:ctrlp\_custom\_ignore = {  \ 'dir': '\.git$\|\.hg$\|\.svn$',  \ 'file': '\.exe$\|\.so$\|\.dll$\|\.pyc$' }    if executable('ag')  let s:ctrlp\_fallback = 'ag %s --nocolor -l -g ""'  elseif executable('ack-grep')  let s:ctrlp\_fallback = 'ack-grep %s --nocolor -f'  elseif executable('ack')  let s:ctrlp\_fallback = 'ack %s --nocolor -f'  " On Windows use "dir" as fallback command.  else  let s:ctrlp\_fallback = 'find %s -type f'  endif  if exists("g:ctrlp\_user\_command")  unlet g:ctrlp\_user\_command  endif  let g:ctrlp\_user\_command = {  \ 'types': {  \ 1: ['.git', 'cd %s && git ls-files . --cached --exclude-standard --others'],  \ 2: ['.hg', 'hg --cwd %s locate -I .'],  \ },  \ 'fallback': s:ctrlp\_fallback  \ }    if isdirectory(expand("~/.vim/bundle/ctrlp-funky/"))  " CtrlP extensions  let g:ctrlp\_extensions = ['funky']  "funky  nnoremap <Leader>fu :CtrlPFunky<Cr>  endif  endif    " complete  autocmd Filetype \* if &omnifunc == "" | setlocal omnifunc=syntaxcomplete#Complete | endif  autocmd FileType css setlocal omnifunc=csscomplete#CompleteCSS  autocmd FileType html,markdown setlocal omnifunc=htmlcomplete#CompleteTags  autocmd FileType javascript setlocal omnifunc=javascriptcomplete#CompleteJS  autocmd FileType python setlocal omnifunc=pythoncomplete#Complete    "进行版权声明的设置  "添加或更新头  map <F4> :call TitleDet()<cr>'s  autocmd BufNewFile \*.py :call AddTitle()  function AddTitle()  call append(0,"#! /usr/bin/env python")  call append(1,"# -\*- coding: utf-8 -\*-")  call append(2,"# vim:fenc=utf-8")  call append(3,"# Copyright © XYM")  call append(4,"# CreateTime: ".strftime("%Y-%m-%d %H:%M:%S"))  call append(5,"")  endf    let python\_highlight\_all = 1 |

## vim相关插件(vimrc.bundles文件)

|  |
| --- |
| if &compatible  set nocompatible  end    filetype off  set rtp+=~/.vim/bundle/vundle  call vundle#rc()  Bundle "gmarik/vundle"  " 星级越多越推荐, 5星为必选，4星强烈推荐，3星比较推荐  " 文件模糊匹配查找 推荐：\*\*\*\*  Bundle "kien/ctrlp.vim"  " 函数模糊匹配查找(,fu) 推荐：\*\*\*\*  Bundle 'tacahiroy/ctrlp-funky'  " 文件索引树 推荐：\*\*\*\*\*  Bundle "scrooloose/nerdtree"  " 自动注释 推荐：\*\*\*\*\*  Bundle 'scrooloose/nerdcommenter'  " 语法检查 推荐：\*\*\*\*  Bundle "scrooloose/syntastic"  " 括号自动匹配 推荐：\*\*\*\*  Bundle 'jiangmiao/auto-pairs'  " 引号处理 推荐：\*\*\*\*  Bundle 'tpope/vim-surround'  " 光标多行编辑 <C-n> <C-x> <C-p> 推荐：\*\*\*\*  Bundle 'terryma/vim-multiple-cursors'  " 配色主题 推荐：\*\*\*\*  Bundle "altercation/vim-colors-solarized"  " 缩进引导线 推荐：\*\*\*\*  Bundle "nathanaelkane/vim-indent-guides"  " 文本同级文字选取，和文本处理 推荐：\*\*\*\*  Bundle 'kana/vim-textobj-indent'  Bundle 'kana/vim-textobj-user'  " 保存vim编辑信息，比如最后的位置 推荐：\*\*\*\*  Bundle "vim-scripts/restore\_view.vim"    " 状态栏效果 推荐：\*\*\*  Bundle "bling/vim-airline"    " python 推荐：\*\*\*\*  Bundle 'yssource/python.vim'  Bundle 'python\_match.vim'  Bundle 'pythoncomplete'  Bundle 'hdima/python-syntax'  Bundle 'hynek/vim-python-pep8-indent'    " js 推荐：\*\*\*\*  Bundle 'elzr/vim-json'  Bundle 'groenewege/vim-less'  Bundle 'pangloss/vim-javascript'  Bundle 'briancollins/vim-jst'  Bundle 'kchmck/vim-coffee-script'    " html 推荐：\*\*\*\*  Bundle 'amirh/HTML-AutoCloseTag'  Bundle 'hail2u/vim-css3-syntax'  Bundle 'gorodinskiy/vim-coloresque'  Bundle 'tpope/vim-haml'  Bundle 'mattn/emmet-vim'    if filereadable(expand("~/.vimrc.bundles.local"))  source ~/.vimrc.bundles.local  endif  filetype on |