# **修改linux下默认为python3版本**



[Qidian213](https://me.csdn.net/u014525760" \t "https://blog.csdn.net/u014525760/article/details/_blank) 2018-03-24 10:37:13 IMG_257 10031 IMG_258 收藏 9

分类专栏： [python](https://blog.csdn.net/u014525760/category_7497020.html" \t "https://blog.csdn.net/u014525760/article/details/_blank) 文章标签： [python](https://www.csdn.net/gather_24/MtjaQg4sNDk0LWJsb2cO0O0O.html" \t "https://blog.csdn.net/u014525760/article/details/_blank)

版权

首先：python 查看默认python 版本

修改默认的版本 并不是删除你不需要的版本，因为系统的许多底层是依赖python的，删除后可能会导致系统无法正常运行。

接着需要做的是，删除/usr/bin目录下的python link文件

sudo rm -rf /usr/bin/python

删除后再建立新的链接关系：

sudo ln -s /usr/bin/python3 /usr/bin/python

如果想还原回原python2.7，只需

sudo rm -rf /usr/bin/python

sudo ln -s /usr/bin/ptyhon2.7 /usr/bin/python

https://blog.csdn.net/u011534057/article/details/51615193

****2、 在系统级修改 Python 版本****

我们可以使用 update-alternatives 来为整个系统更改 Python 版本。以 root 身份登录，首先罗列出所有可用的 python 替代版本信息：

****[python]**** [view plain](https://blog.csdn.net/u011534057/article/details/51615193" \o "view plain) [copy](https://blog.csdn.net/u011534057/article/details/51615193" \o "copy)

1. # update-alternatives --list python
2. update-alternatives: error: no alternatives **for** python

如果出现以上所示的错误信息，则表示 Python 的替代版本尚未被 update-alternatives 命令识别。想解决这个问题，我们需要更新一下替代列表，将 python2.7 和 python3.4 放入其中。

****[python]**** [view plain](https://blog.csdn.net/u011534057/article/details/51615193" \o "view plain) [copy](https://blog.csdn.net/u011534057/article/details/51615193" \o "copy)

1. # update-alternatives --install /usr/bin/python python /usr/bin/python2.7 1
2. update-alternatives: using /usr/bin/python2.7 to provide /usr/bin/python (python) **in** auto mode
3. # update-alternatives --install /usr/bin/python python /usr/bin/python3.4 2
4. update-alternatives: using /usr/bin/python3.4 to provide /usr/bin/python (python) **in** auto mode

如何查看当前python的工作目录呢，只需要按如下操作就可以

1. **import** sys
2. sys.executable

'/root/anaconda3/bin/python'

--install 选项使用了多个参数用于创建符号链接。最后一个参数指定了此选项的优先级，如果我们没有手动来设置替代选项，那么具有最高优先级的选项就会被选中。这个例子中，我们为 /usr/bin/python3.4 设置的优先级为2，所以update-alternatives 命令会自动将它设置为默认 Python 版本。

****[python]**** [view plain](https://blog.csdn.net/u011534057/article/details/51615193" \o "view plain) [copy](https://blog.csdn.net/u011534057/article/details/51615193" \o "copy)

1. # python --version
2. Python 3.4.2

接下来，我们再次列出可用的 Python 替代版本。

****[python]**** [view plain](https://blog.csdn.net/u011534057/article/details/51615193" \o "view plain) [copy](https://blog.csdn.net/u011534057/article/details/51615193" \o "copy)

1. # update-alternatives --list python
2. /usr/bin/python2.7
3. /usr/bin/python3.4

现在开始，我们就可以使用下方的命令随时在列出的 Python 替代版本中任意切换了。

****[python]**** [view plain](https://blog.csdn.net/u011534057/article/details/51615193" \o "view plain) [copy](https://blog.csdn.net/u011534057/article/details/51615193" \o "copy)

1. # update-alternatives --config python
2. # python --version
3. Python 2.7.8