**0 - mie note:**

I link forniti da github sono parzialmente incorretti/imprecisi o non completi.

Questa guida dovrebbe fare le correzioni necessarie, ma non posso essere sicuro di questa cosa.

Questa guida funziona per Ubuntu, per altri SO seguire i link (che ho usato anche io)

1 - https://github.com/colesbury/nogil?tab=readme-ov-file#install-with-pyenv

2 - https://github.com/colesbury/nogil/wiki/Install-nogil-with-pyenv-on-Linux

3 - https://github.com/pyenv/pyenv/wiki#suggested-build-environment

4 - <https://www.programmareinpython.it/blog/pyenv-python-version-manager-linux-mac-os/>

**1 - INSTALLAZIONE PYENV, che gestirà l'installazione del nuovo interprete python**

seguo da link 3 e 4

1.1 : dipendenze

Lancia da terminale:

sudo apt update; sudo apt install build-essential libssl-dev zlib1g-dev \

libbz2-dev libreadline-dev libsqlite3-dev curl \

libncursesw5-dev xz-utils tk-dev libxml2-dev libxmlsec1-dev libffi-dev liblzma-dev

1.2 : installazione pyenv e aggiunta del PATH

facendo attenzione all’output dello script:

curl https://pyenv.run | bash

l’output indicherà delle righe di codice da inserire in fondo a due file testuali (nascosti).

Aprire i file con nano o xdg-open e inserire le righe come richiesto.

1.3 : per continuare

CHIUDERE LA SHELL PER CONTINUARE, per vedere il path effettivamente aggiunto

**2 - installazione Python nogil e settaggio ambiente**

aeguo da link 1 e 2

Lancia:

pyenv update

poi

pyenv install nogil-3.9.10-1

atteso qualche minuto e terminata l’installazione, sarà possibile vedere i dettagli nella cartella “/home/user/.pyenv/versions/nogil-3.9.10-1” .

Già ora si potrebbe lanciare il comando /home/user/.pyenv/versions/nogil-3.9.10-1/bin/python3.9

che permette di lanciare l’interprete (un buon metodo per verificare che l’installazione sia andata a buon fine, siccome al lancio l’interprete indicherà la versione e il fatto che è una release “nogil”).

Tuttavia ci serve un metodo più comodo per lanciare l’interprete o includerlo nei nostri script

2.1 : prima alternativa per l’esecuzione

L’alternativa indicata dal canale github ufficiale è di lanciare un comando (indicato in fondo) per impostare l’interprete python come “globale”.

Questo fa si che al lancio del comando “python” da terminale, venga aperto in automatico l’interprete nogil.

Non volendo installare di default una versione potenzialmente instabile sul mio dispositivo, preferisco effettuare l’operazione a mano.

Questo mi permetterà di scegliere il nome per questo interprete, inserendone uno più indicato e significativo.

Non evidenzio il comando perché non ne consiglio il lancio, nei casi come il mio, ma lo riporto per completezza:

“pyenv global nogil-3.9.10-1” .

2.2 : inserisco un link nella cartella “bin”

Perché inserire un link nella cartella bin?

Quando viene lanciato un eseguibile da terminale, il S.O cerca in un insieme di percorsi contenuti in una variabile detta “PATH”, se non si specifica la directory.

Per far sì che questa versione di python sia utilizzabile da qualsiasi directory del filesystem, c’è bisogno di inserirla nel PATH. Una delle cartelle che contenuta in PATH è proprio la cartella /bin , dove andremo a creare un link all’eseguibile. (alternativamente si poteva inserire la directory fra quelle presenti in PATH)

L’ eseguibile era stato installato da pyenv nella cartella: ~/.pyenv/versions/nogil-3.9.10-1/bin . Chiameremo questo link python\_nogil, che corrisponderà anche al nome che servirà per eseguire questo nuovo interprete.

Il comando per effettuare quest’operazione è

sudo ln ~/.pyenv/versions/nogil-3.9.10-1/bin/python3.9 /bin/python\_nogil .

Adesso, da qualsiasi directory, potremo lanciare questa versione con il comando

python\_nogil (CTRL+D per uscire). Alternativamente potremo indicarlo in prima riga in uno script python come shebang.

COMPLETARE

**=============================================**

**1 - l**

l

1.1 : l

l