
# UL HPC School

## [Advanced] Prototyping with Python

**UL High Performance Computing (HPC) Team**

**C. Parisot**

University of Luxembourg (UL), Luxembourg

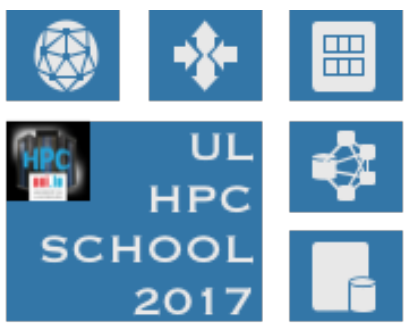
http://hpc.uni.lu

## Latest versions available on Github:

UL HPC tutorials: https://github.com/ULHPC/tutorials

UL HPC School: http://hpc.uni.lu/hpc-school/

Tutorial sources: https://github.com/ULHPC/tutorials/tree/devel/advanced/Python/




# Summary

# Main Objectives of this Session

- Run Python code on the cluster
- Install and use your own **Python packages**
  - ↪ Create a **virtual environment** to use several version of the same package
- Compile your code in **C** to have better performances
- Use **Scoop** to distribute your code on the cluster

# Summary

# Python / Pip

- `pip`: Python package manager
  - ↪ "nice" python packages: `mkdocs`...
  - ↪ Windows: install via Chocolatey

```
$> pip install <package>                    # install <package>
```

# Python / Pip

- `pip`: Python package manager
  - ↪ "nice" python packages: `mkdocs`...
  - ↪ Windows: install via Chocolatey

```
$> pip install <package>                    # install <package>
```

```
$> pip install -U pip                       # upgrade on Linux/Mac OS
```

# Python / Pip

- `pip`: Python package manager
  - ↪ "nice" python packages: `mkdocs`...
  - ↪ Windows: install via Chocolatey

```
$> pip install <package>                    # install <package>
```

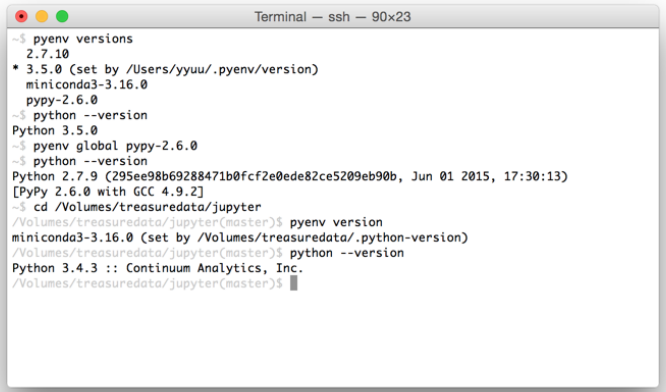
```
$> pip install -U pip                       # upgrade on Linux/Mac OS
```

- Dump python environment to a requirements file

```
$> pip freeze -l > requirements.txt         # as Ruby Gemfiles
```

# Pyenv / VirtualEnv / Autoenv

- pyenv: ≃ RVM/rbenv for Python
- virtualenv ≃ RVM Gemset
- (optional) autoenv
  - ↪ Directory-based shell environments
  - ↪ easy config through .env file. **Ex**:




```
# (rootdir)/.env : autoenv configuration file
pyversion=‘head .python-version‘
pvenv=‘head .python-virtualenv‘

pyenv virtualenv --force --quiet ${pyversion} ${pvenv}-${pyversion}
# activate it
pyenv activate ${pvenv}-${pyversion}
```

# Summary

# Virtualenv

- Install `virtualenv` on the cluster using `pip`
- Create your own **virtual environment** to install packages inside it

# Use several version of Python

There are **several versions** of Python available on the cluster.
They have been build against **several toolchains**.
The goal of this part is to compare the different versions available on the cluster.

# Scoop / Cython

Optimize your code for execution on the HPC cluster

- parallelisation using Scoop
- compile your Python code in C for faster execution with Pythran or Cython
- use Numpy package to optimize your code

# Questions?

http://hpc.uni.lu

**The UL High Performance Computing (HPC) Team**
University of Luxembourg, Belval Campus:
Maison du Nombre, 4th floor
2, avenue de l'Université
L-4365 Esch-sur-Alzette
*mail:* hpc@uni.lu