

Python for Data Science: SW01

Script Language
Basic Data Types
Mathematical Operations

Information Technology

February 20, 2025



Content

- Administrative / Introduction
 - Course material
 - Python: interpreter vs. compiler language
- Intelligent Development Environment (IDE)
 - PyCharm installation
 - PyCharm introduction with local project
 - Python console vs. script
 - Virtual development environment: venv
- Python Basics
 - Variables and basic data types
 - Mathematical operations

Main goals

PyCharm installed and running | First Python exercises

Administrative / Introduction

Information Technology

February 20, 2025

Teaching team



Andreas Melillo

Dozent

andreas.melillo@hslu.ch



Ramón Christen

Sen. Wissenschaftlicher Mitarbeiter /
Doktorand

ramon.christen@hslu.ch



Tim Giger

Wissenschaftlicher Mitarbeiter

tim.giger@hslu.ch

Course material

Our lecture is based on:

Einführung in Python 3 (4. Auflage) or **Python Tutorial** (english version)

Bernd Klein

Hanser Verlag

ISBN: 978-3-446-46379-0

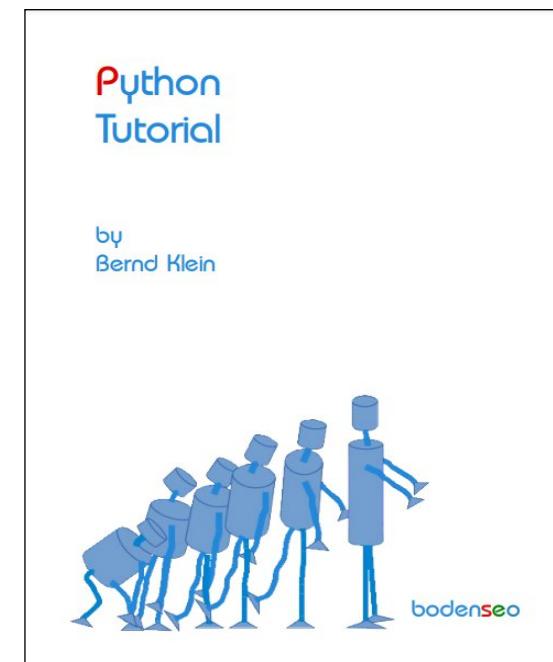
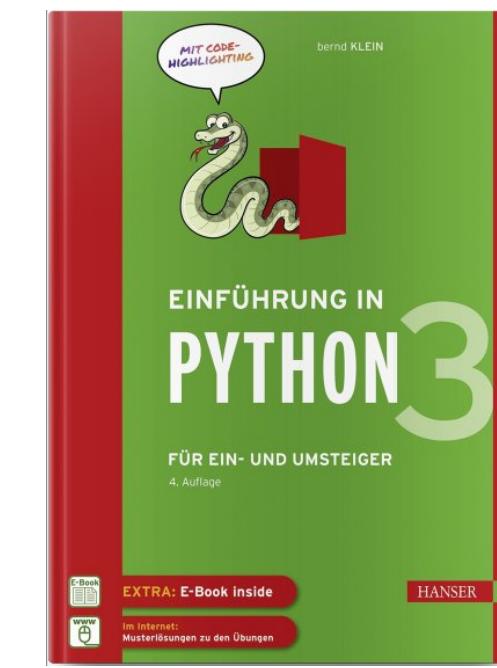
URL:

- <https://www.python-kurs.eu/> (german)
- <https://www.python-course.eu/> (english)
- https://www.python-course.eu/bernd_klein_python_tutorial_a4.pdf (english)

Online course on tutorials.eu (coupon only valid for a few days)

- **The Complete Python 3 Masterclass – From Beginner to Pro**

https://academy.tutorials.eu/p/the-complete-python-3-masterclass-from-beginner-to-pro-live?coupon_code=HSLUSS252&product_id=5867708



Course material

Official Python documentation

<https://docs.python.org/3/>

Ad-hoc exercises on ilias

PDS course container > section “Exercises”

Cheat sheets

Linux: on ilias in PDS course container > section “Supporting Material”

Python: official Python beginner's guide <https://wiki.python.org/moin/BEGINNERSGUIDE>



python™

» BeginnersGuide

» BeginnersGuide

[FRONTPAGE](#) »
[RECENTCHANGES](#) »
[FINDPAGE](#) »
[HELPCONTENTS](#) »
BEGINNERSGUIDE »

Page

» Immutable Page
» Comments
» Info
» Attachments
» More Actions: ▾

User

» Login

Beginner's Guide to Python

New to programming? Python is free and easy to learn if you know where to start! This guide will help you get started.

[Chinese Translation/中文版入门](#)

New to Python?

Read [BeginnersGuide/Overview](#) for a short explanation of what Python is.

Getting Python

Next, install the Python 3 interpreter on your computer. This is the program that reads Python programs before you can do any Python programming. Mac and Linux distributions may include an outdated version of Python. See [BeginnersGuide/Download](#) for instructions to download the correct version.

There are also Python interpreter and IDE bundles available, such as  [Thonny](#). Other options include [PyCharm](#), [Jupyter Notebook](#), and [Spyder](#).

At some stage, you'll want to edit and save your program code. Take a look at [HowToEditPython](#).

Learning Python

PDS - LINUX CHEATSHEET	
ERWIN MATHIS, SIMON BRODA AND RAMÓN CHRISTEN UNIVERSITY OF APPLIED SCIENCES AND ARTS - HSLU	
<h3>SYSTEM AND ACCOUNTS</h3> <p>System Administration</p> <pre>!! repeat last command exit close current session (terminal) sudo execute command as root user uname -a show system information passwd set password of current user passwd user set password of user <i>user</i> * reboot immediate system reboot * kill proc kill process <i>proc</i></pre> <p>Processes</p> <pre>/bin execute binary <i>bin</i> in current directory ps list snapshot of running processes top list running processes man cmd show manual for command <i>cmd</i> which cmd locate command (bin) <i>cmd</i> whereis cmd locate bin, src and man for command <i>cmd</i></pre>	<h3>NAVIGATION AND CONTENT</h3> <p>Navigation</p> <pre>cd change directory to <i>home</i> folder cd ~ cd to home folder cd . cd one level up cd ../.. cd two levels up cd - cd to previous dir cd dir1/dir2 cd to directory <i>dir1</i>/<i>dir2</i> pwd show current/working directory</pre> <p>List Content</p> <pre>ls list directory content ls -l list content details ls -a list all (inc. hidden files) ls -la list details of all content ls -l .txt list details of .txt files</pre>
<h3>DEBIAN (UBUNTU) PACKAGE MANAGER</h3> <p>Search and Version Control</p> <pre>apt-cache search <i>pkg</i> 1. list packages matching term <i>pkg</i> apt search <i>pkg</i> 2. remark installed packages list packages matching term <i>pkg</i></pre> <p>Install and Update</p> <pre>apt-get update apt-get upgrade apt-get upgrade <i>pkg</i> apt-get install <i>pkg</i> apt install <i>pkg</i>*</pre>	<h3>FILES AND FOLDERS</h3> <p>Download Files</p> <pre>curl https://....csv download file from source https://....csv wget https://....csv download file from source https://....csv</pre> <p>Copy - Move - Rename - Remove</p> <pre>cp <i>f1</i> <i>f2</i> copy file <i>f1</i> to file <i>f2</i> cp -f <i>f1</i> <i>path</i> copy folder <i>f1</i> to <i>path</i> mv <i>f1</i> <i>f2</i> move (rename) file <i>f1</i> to file <i>f2</i> rm <i>f1</i> remove file <i>f1</i> rm -rf <i>f1</i> remove folder <i>f1</i></pre> <p>File and Folder Handling</p> <pre>find -name <i>fn</i> find file/folder by name <i>fn</i> touch <i>fi</i> create file <i>fi</i> mkdir <i>dir</i>/<i>subdir</i> make directory <i>dir</i>/<i>subdir</i> chmod <i>fn</i> change access rights of file/folder <i>fn</i> chmod u+x <i>fn</i> add exec. rights to user for file/folder <i>fn</i> cat <i>fi</i> print content of file <i>fi</i> head -n <i>fi</i> print x first lines of file <i>fi</i> tail -n <i>xi</i> print x last lines of file <i>fi</i> vim <i>fi</i> open file/folder <i>fn</i> in vim editor</pre>
<p>* commands have to be executed as root user (i.e. sudo <i>cmd</i>)</p>	<h3>NETWORK</h3> <p>Basic Network Commands</p> <pre>ssh <i>usr</i>@<i>host</i> set-up ssh connection to <i>host</i> for user <i>usr</i> exit close current ssh session ping -c <i>x</i> <i>host</i> ping <i>x</i> times to <i>host</i></pre> <p>JUPYTER NOTEBOOK SERVER</p> <p>Server Admin</p> <pre>jupyter-notebook start jupyter notebook server jupyter-notebook list list running notebook servers jupyter-notebook stop 8888 stop server on port 8888 kill \$(pgrep jupyter) kill all jupyter processes</pre> <p>PYTHON</p> <p>Python Packages</p> <pre>pip list list installed python packages pip search <i>pkgs</i> search for package name <i>pkgs</i> pip install <i>pkgs</i> install pip package <i>pkgs</i> python -V show the version of python interpreter python -m venv <i>path</i> create virtual environment in <i>path</i></pre> <p>VIRTUAL ENVIRONMENT</p> <p>Admin Virtual Environment</p> <pre>. activate activate virtual environment source activate activate virtual environment</pre>

Before we start

Comment on course content:

- The course content is currently rebuilt
- Agenda (-> ILIAS) for the second half of the semester may still change

Comment on typical lecture plan:

- Start with theoretical inputs, with live coding examples
- End with exercise sessions
- Passive streaming on onsite lectures

Comment on inhomogeneous backgrounds:

- How many of you have experience with programming?
- How many of you have experience with python?
- We try our best to keep the lessons interesting for everybody!

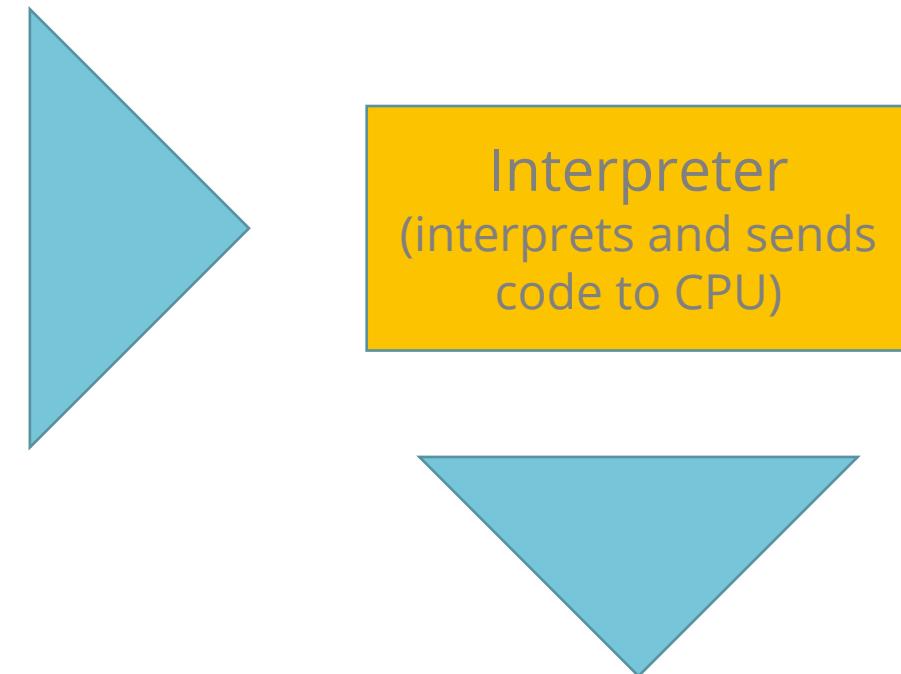


Python: interpreter vs. compiler language

Interpreter (script) Language

- Ex: Bash, JavaScript, Matlab, Python, R, Ruby

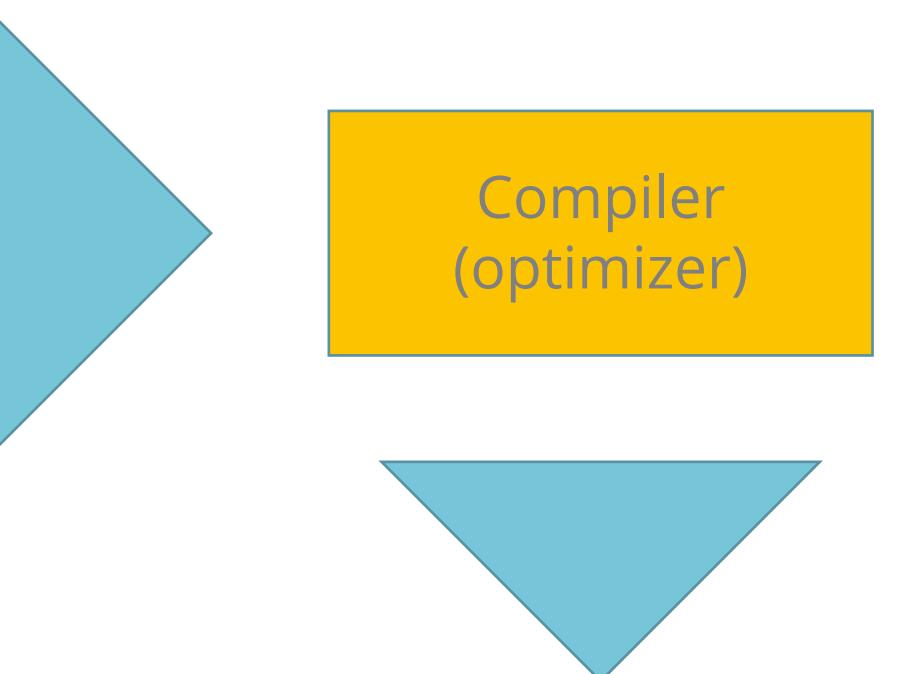
Human interpretable source code



Compiler Language

- ## • C, C++, C#, Java

Human interpretable source code



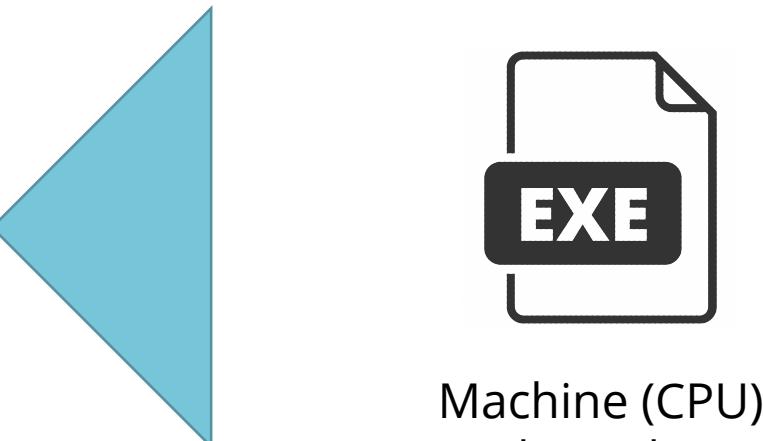
Compiler (optimizer)

```
File Edit View Terminal Tabs Help
=====
Netrw Directory Listing                               (netrw v173)
/home/prypjat/hslu.cloud/hslu-rch
 Sorted by      name
 Sort sequence: [V]$, \<core\%(\.\d+)\=\>, \.h$, \.c$, \.cpp$, \-\*=*$,*\.\$,
 Quick Help: <f1>:help -:go up dir D:delete R:rename s:sort-by x:specia
=====
./
/
oclab/
lecture/
misc/
Ohd/
projects/
otero_lib/
.
.
.

No matching autocommands: FileType netrw          13,1           All
```

Console / UI

Console / UI



Machine (CPU) code package

Python: interpreter vs. compiler language

Interpreter (script) Language

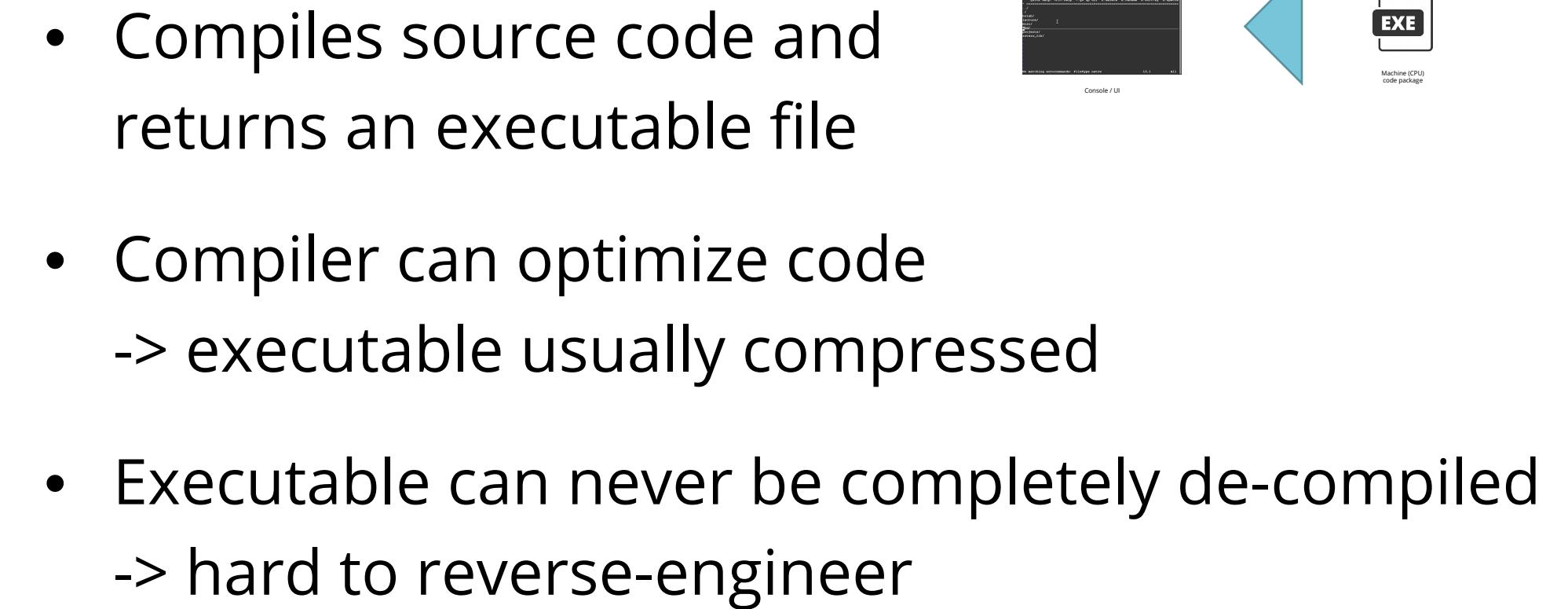
- Ex: Bash, JavaScript, Matlab, Python, R, Ruby



- Code **line by line** sent to interpreter
- Interpreter provides console
 - > ad-hoc commands
- No optimization
- Slow in execution

Compiler Language

- C, C++, C#, Java



Intelligent Development Environment IDE

Information Technology

February 20, 2025

Intelligent Development Environment (IDE)

Intelligent Development Environments (IDE) typically comprise supporting additional tools for convenient programming.

Provided tools may include:

- Debugger
- Deployment chain control
- File browser
- Spell checker
- Terminal
- Todo list
- Remote development
- Remote execution
- Version control interface

Alternatives:

- Any text editor



Python console vs. script

Python is an interpreter language. That means, every single instruction is sent **sequentially** to the interpreter.

Python **console** is the direct interface (connection) to the interpreter.

- Sending a command in the console to the interpreter, immediately returns the result on the next line.

A Python **script** is a collection of commands.

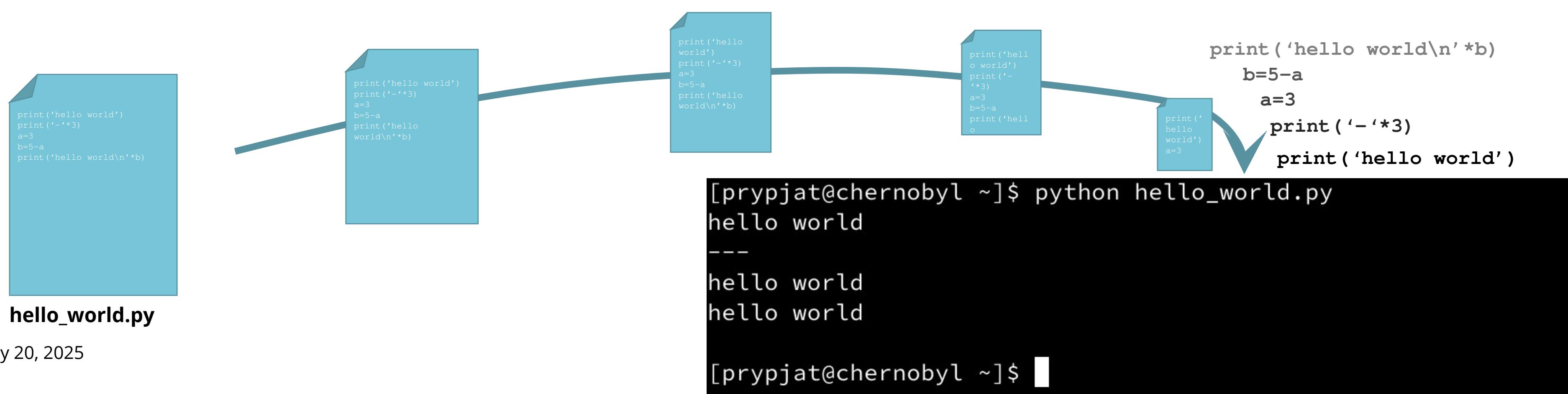
- Executing a Python script means, sending all contained commands consecutively to the interpreter.

```
[prypjat@chernobyl ~]$ python
Python 3.13.1 (main, Dec 4 2024, 18:05:56) [GCC 14.2.1 20240910] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> print('hello world')
hello world
>>> █
```

Command sent to interpreter

Reply to sent command

Ready for next command



PyCharm installation

PyCharm is provided by JetBrains and available in two editions:
(for Mac, Windows or Linux)

1. Community edition

- free license
- limited functionality

2. Professional edition

- free for student and lecturer with annual education license
- supports remote app development
- Jupyter Notebook integration

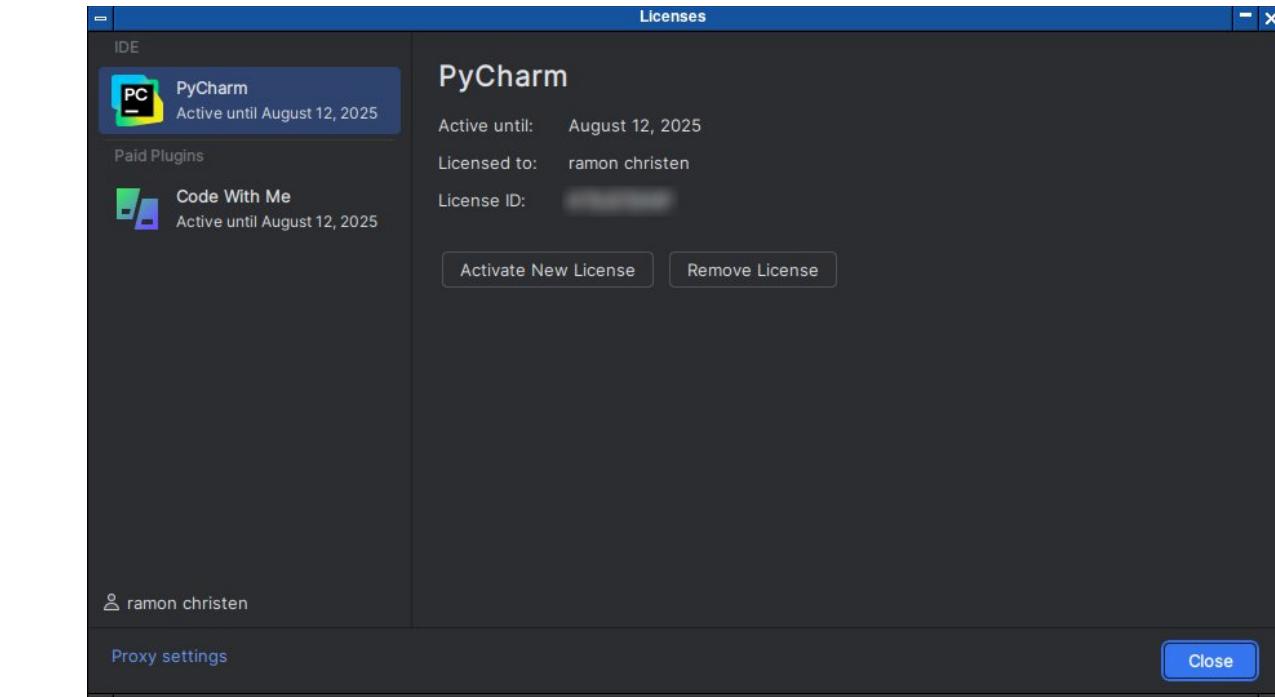
Installation of professional edition (including Python interpreter):

1. download PyCharm professional from <https://www.jetbrains.com/>

for Linux also available in snap: `sudo snap install pycharm-professional`

2. install IDE as any other regular app

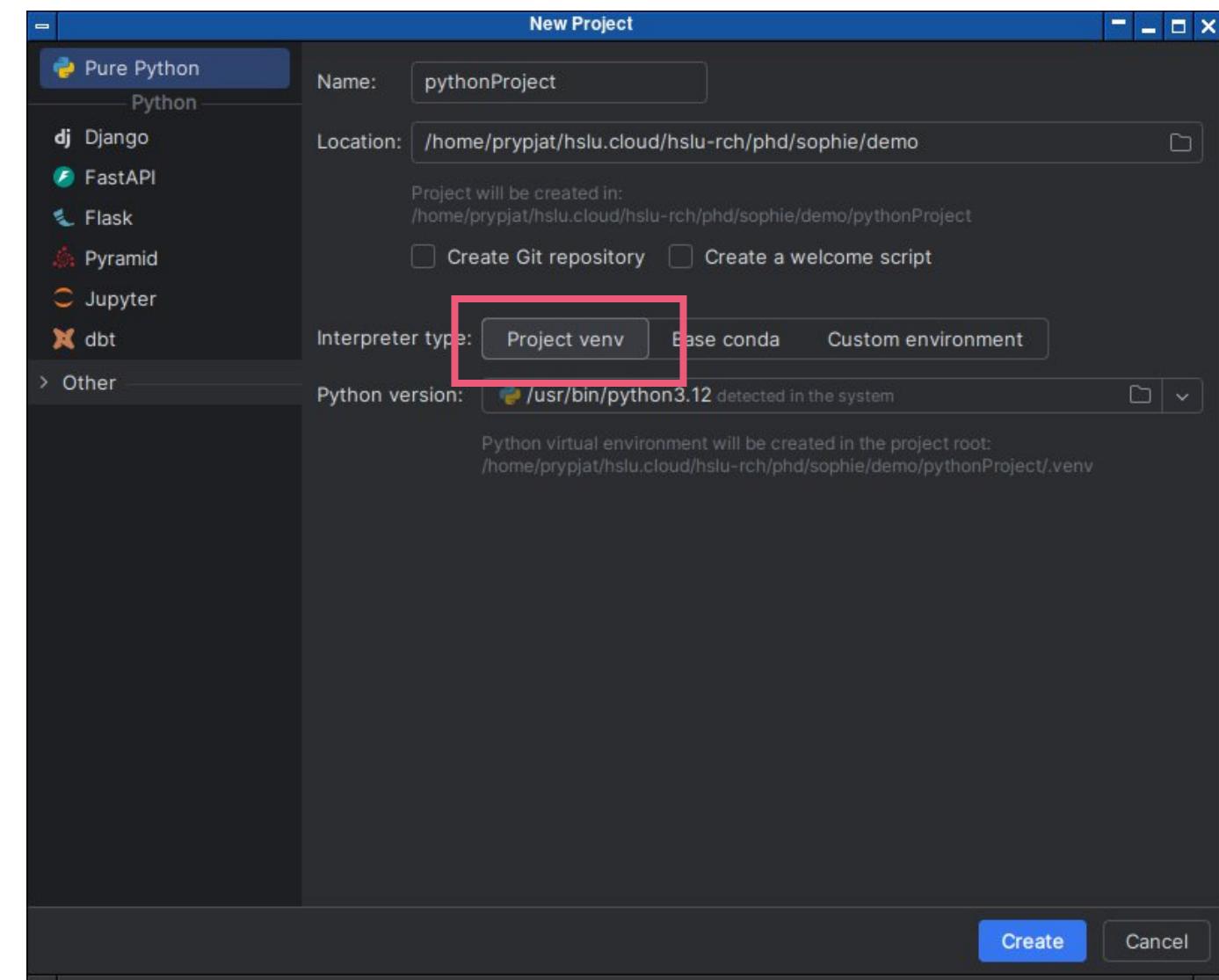
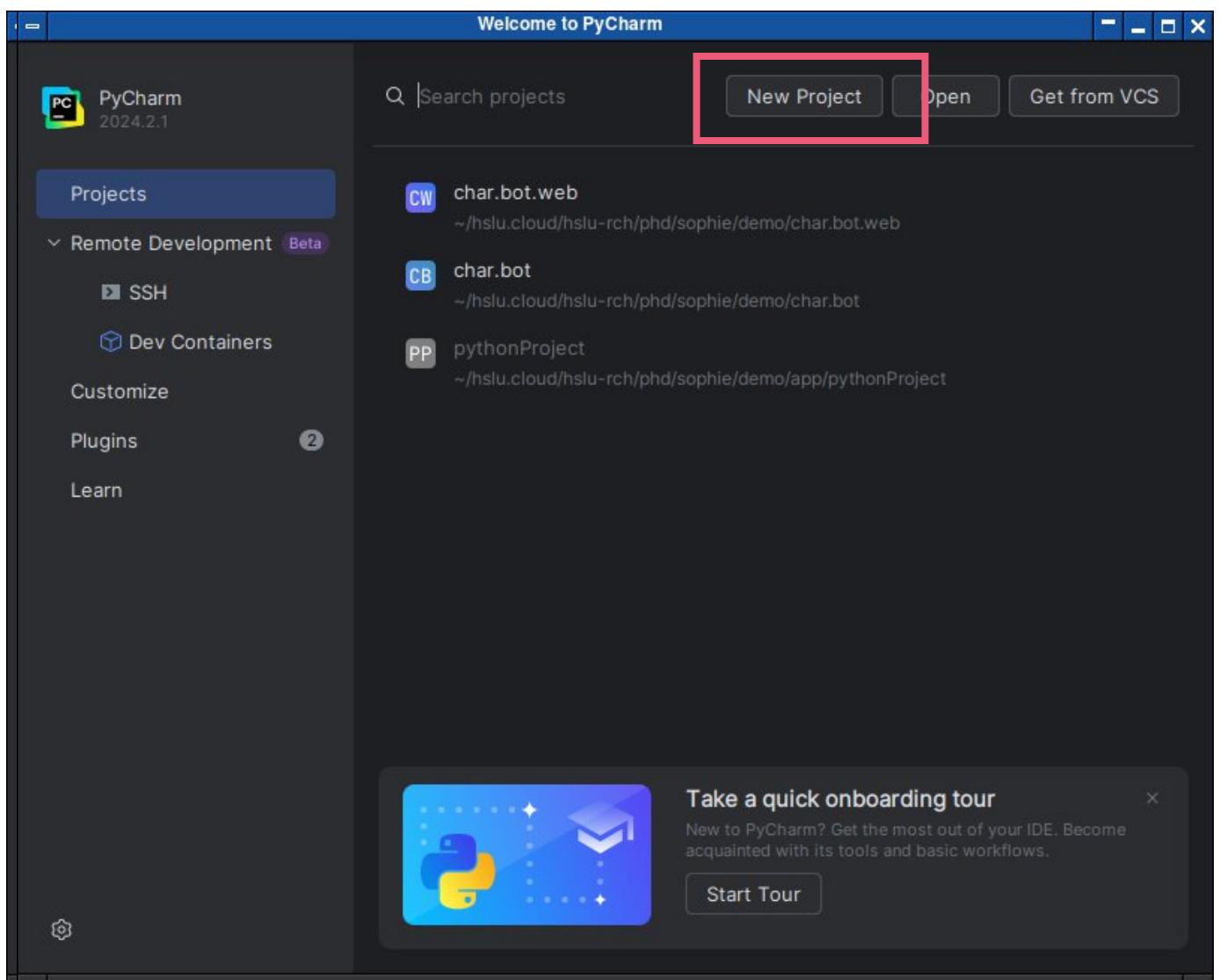
3. create JetBrains education account with university's mail address by registering your app
(without an open project: settings (gear on bottom left) > Manage Licenses... | else: Menu > Help > Register...)



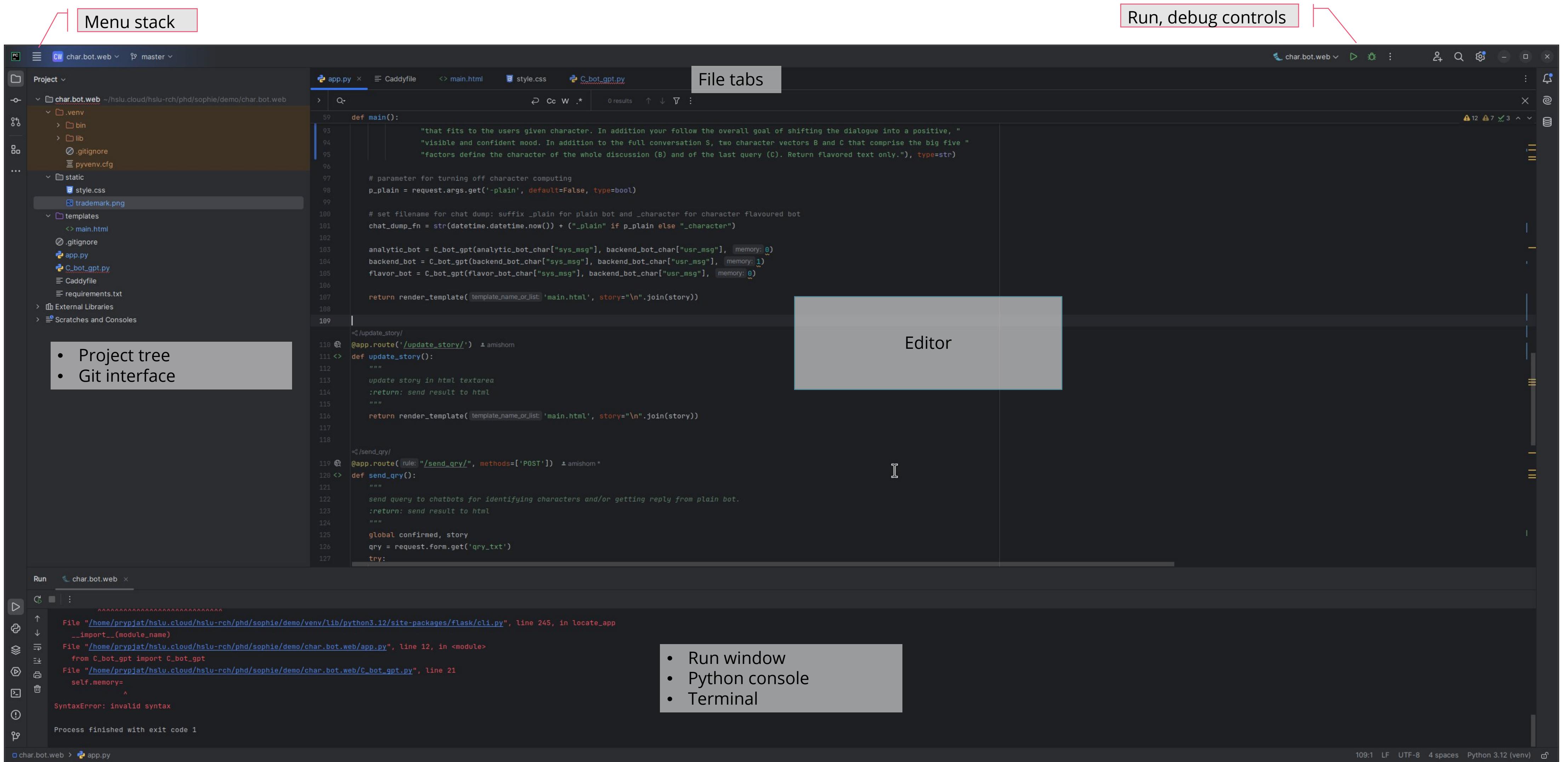
PyCharm introduction with local project

1. Start PyCharm
2. Create New Project

1. Set appropriate project name
2. Select project location
3. Choose project venv (virtual environment will be created in project root)



PyCharm introduction with local project



PyCharm introduction with local project

First “hello world” with Python:

1. Switch to the console window at the bottom
2. Activate the terminal
3. Start the Python console
4. Enter following print command:

```
print("hello world")
```

The screenshot shows the PyCharm IDE's terminal window. The tab bar at the top has 'Terminal' selected, followed by 'Local' and a '+' button. The terminal content is as follows:

```
Terminal Local + 
(venv) [prypjat@chernobyl char.bot.web]$ python
Python 3.12.5 (main, Aug  9 2024, 08:20:41) [GCC 14.2.1 20240805] on linux
Type "help", "copyright", "credits" or "license" for more information.

>>> print("hello world")
hello world
>>>
```

Annotations with red boxes and arrows point to specific parts of the interface:

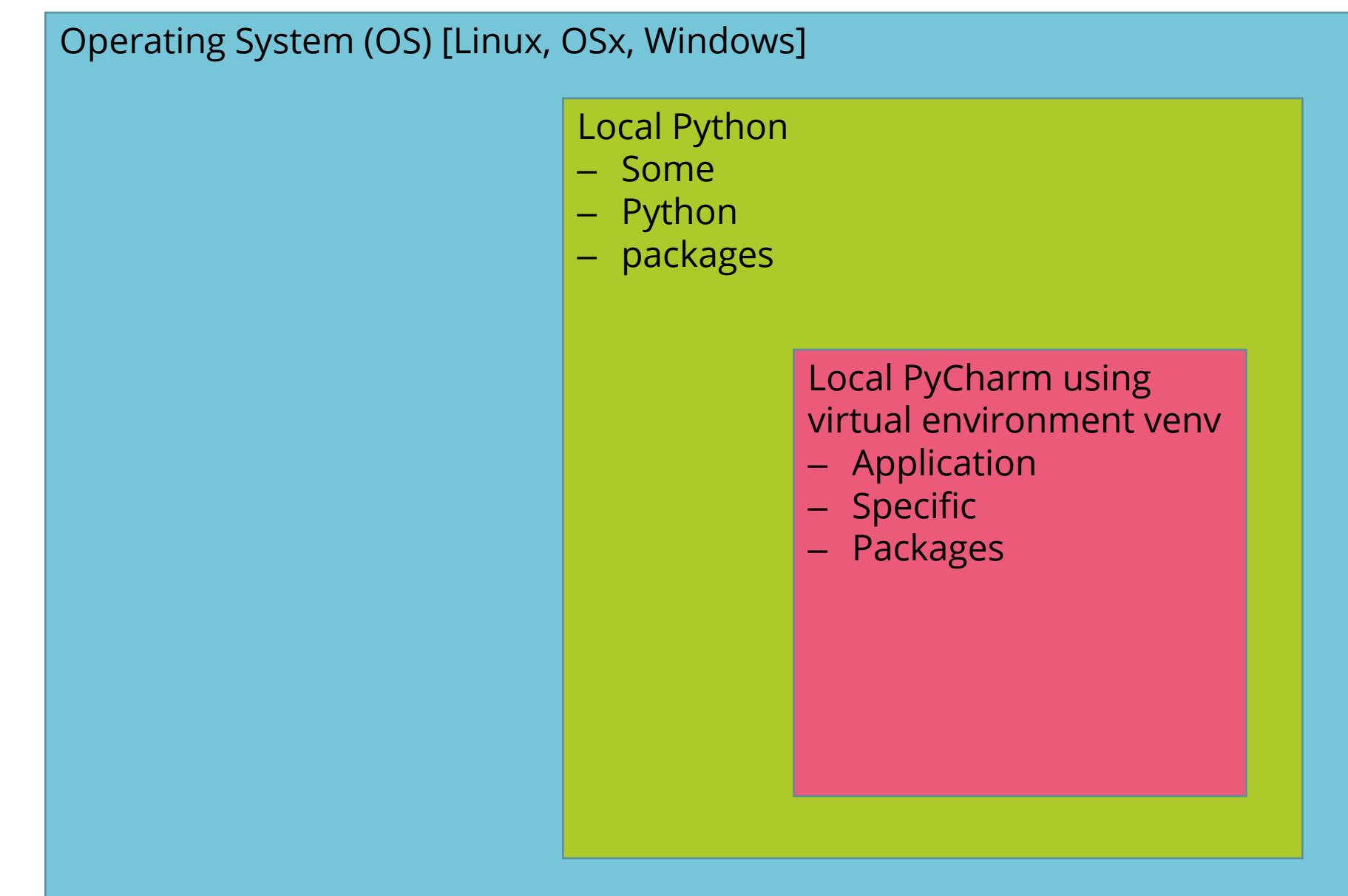
- A box labeled "Terminal window" points to the terminal icon in the bottom dock.
- A box labeled "Start Python console" points to the "python" command in the terminal.
- A box labeled "Python command" points to the "print("hello world")" line in the terminal.
- A box labeled "Python output" points to the "hello world" output line in the terminal.

Virtual development environment: venv

- Python comes with some built-in packages and functions e.g. `print()`, `abs()`, `str()`, `open()`
- Typically, applications do not completely rely on built in functions but refer to external packages.
- External packages can be installed by the Package Installer for Python (pip).
- For avoiding dependency issues and keeping development environment clean, developer program applications in virtual environments (venv).

Create virtual environment using Python:

```
Python3 -m pip venv venvName
```



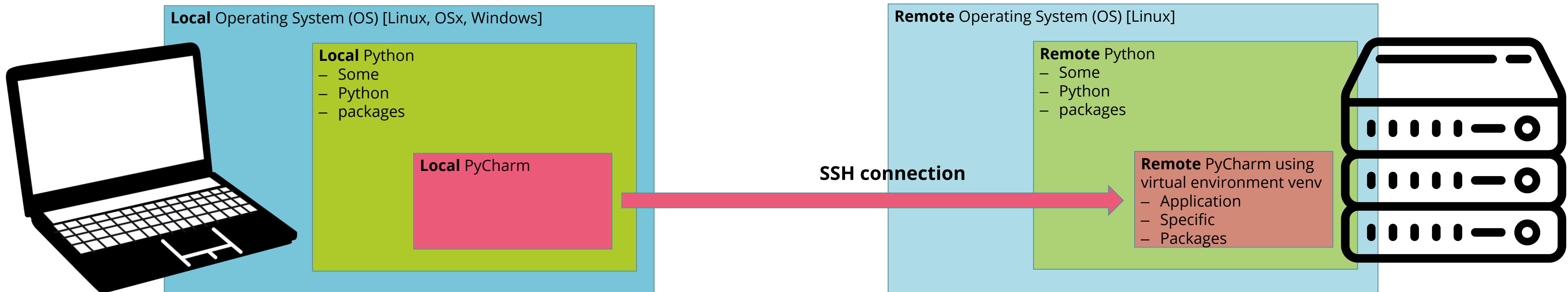
Pycharm for remote application development

Data scientists often deal with large data sets and computationally heavy machine learning algorithms.

Consequently:

- Data sets are stored on remote server.
- Python scripts have large execution times (days, weeks, month).
- Expensive computer resources (GPU, RAM, SSD) are shared in virtualized machines.
- Python scripts are developed on remote systems.

PyCharm allows for remote app development using secure shell (ssh) connection.



PyCharm for remote application development

Setup for remote application development:

1. Ensure the remote machine is up and running. For this course:

- Go to spark.switch.ch
- Login with HSLU switch account
- Enter your personal server IPv4 (86.119.xx.xx) and password (e.g. bd19a4)
- Start your machine

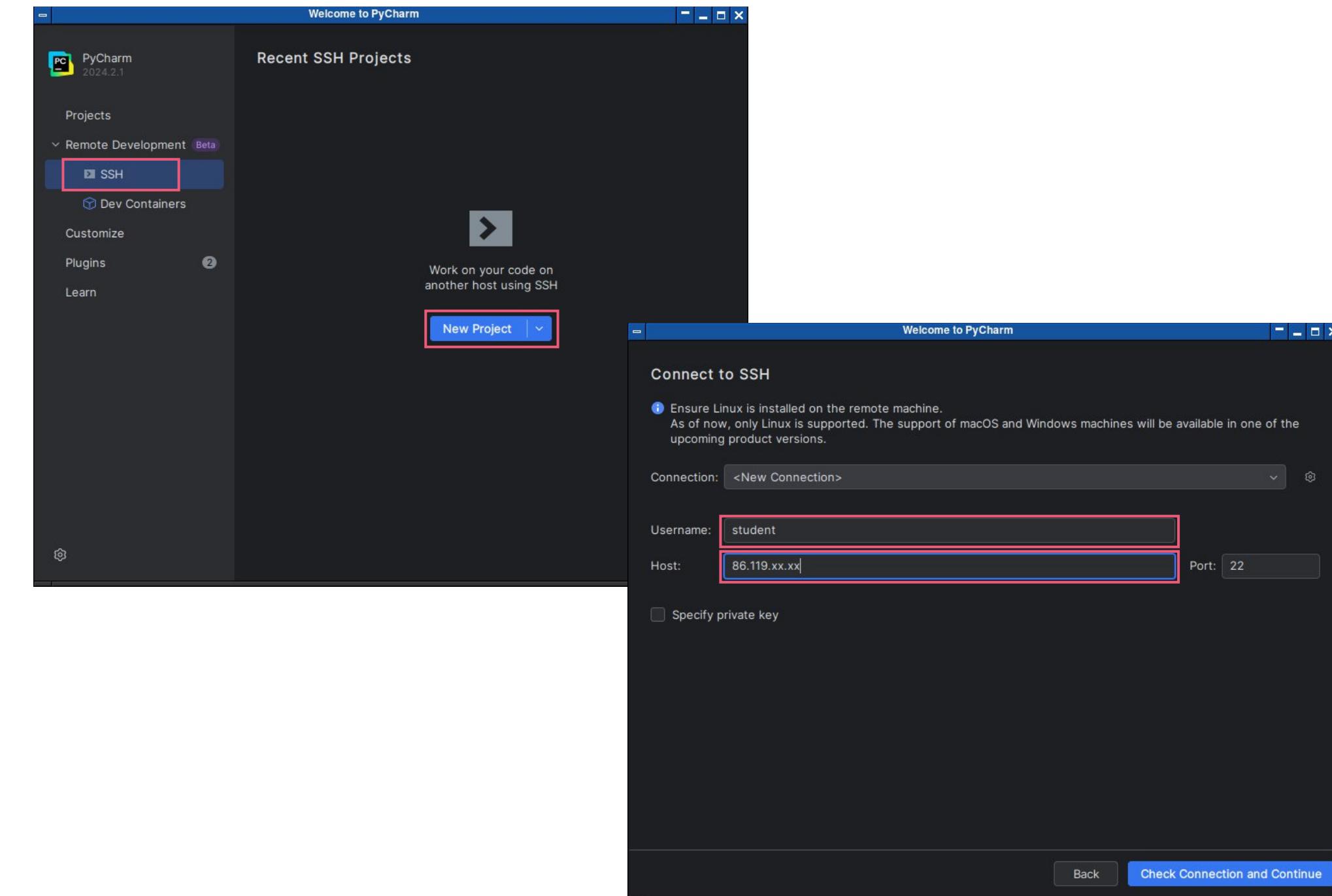
2. Start PyCharm

- Close open project (Menu > File > Close Project)
- Select SSH in Remote Development (left pane)
- Click on “New Project”

• Select connection (or create a new one)
> enter username: **student**
> enter host: **86.119.xx.xx**

- Click on “Check Connection and Continue”

3. Select Python interpreter in: /home/student/venv



Wrap up setup process

- PyCharm Professional: installed
- First python project: created with dedicated virtual environment
- Virtual linux machine: started and connected to
- Remote development: project set up within PyCharm

-> Now we can start to python!

Python Basics

Information Technology

February 20, 2025

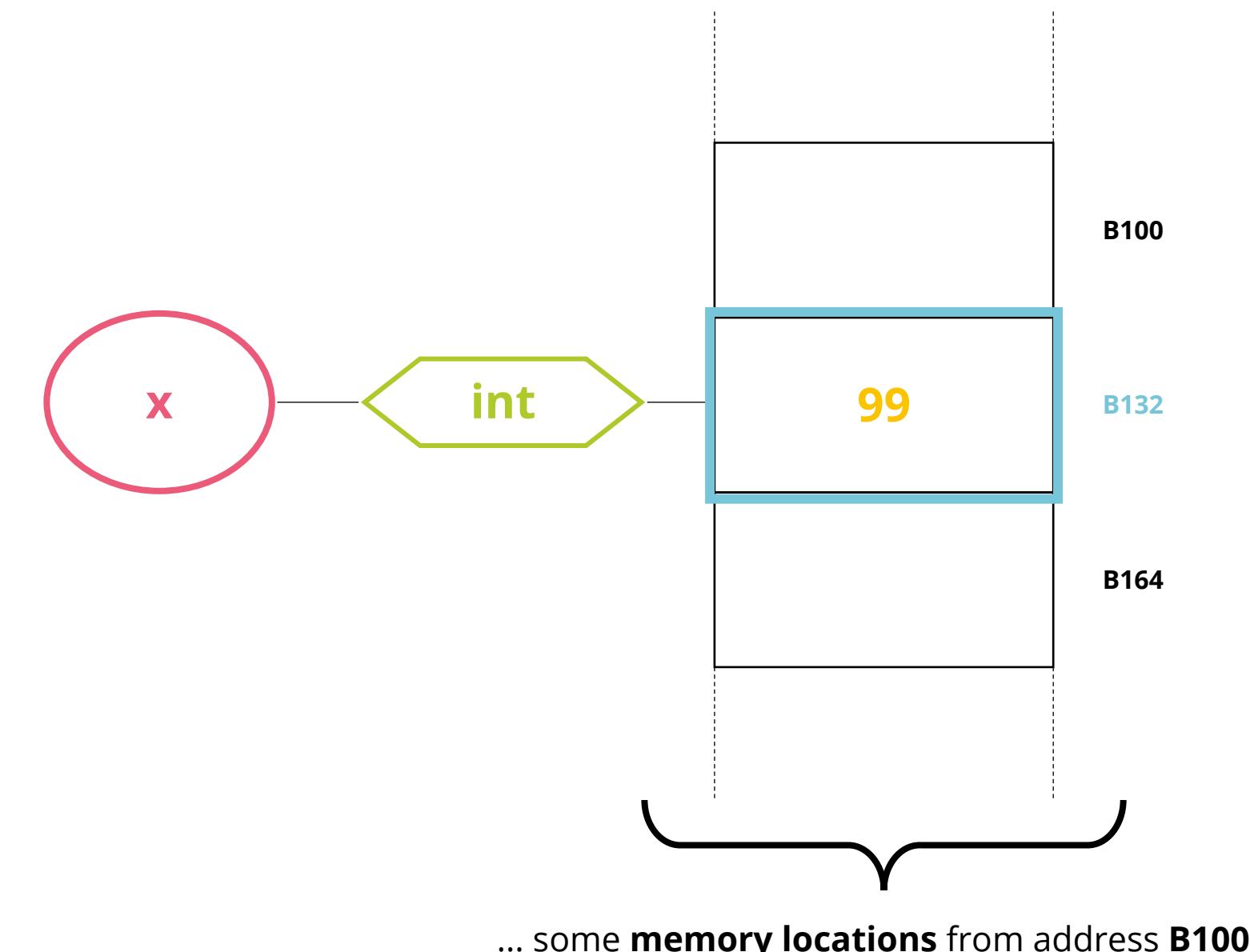
Variables and basic data types

Variables are names referring to a particular or undefined value.

- In math we know them from equations such as:
 $18 = 2 * x$ > x represents a particular value or
 $a^2 = b^2 + c^2$ > a, b and c can be arbitrary values holding the equation.
- In programming languages we use variables as **reference** to a particular storage location.

In Python, variables always consist of:

- **Name**
- **Data type**
- **Storage location**
- **Value**



Variables and basic data types

Python does **not** require any **type** for variable definitions. It assumes the type from the value.

Python	C, C# or Java
<code>x = 99</code> (<code>name = value</code>)	<code>int x = 99</code> (<code>type name = value</code>)

Variable names...

- **have to** start with a letter [a-z,A-Z] (or in special cases with an underscore "_").
- are case sensitive.
- start with lower cases (convention).
- with multiple words are separated with an underscore "hello_world".
-> Camel case (e.g. helloWorld) is not really used (allowed).

Variables and basic data types

Basic data types known in Python:

- Integer (int) 1, 2, 3, ... 100, ... 87293
- Floating point numbers (float) 3.1415962535898
- Strings (str) "hello 'my' world" or 'hello "my" world'
- Boolean (bool) True, False
- Complex numbers 5 + 3j

List types

- List [1, "hello", 3] ->
simple collection of elements, different types possible
- Tuple (1, "hello", 3) ->
difference to list: tuples are immutable
- Dictionary {"a":1, "b":"hello", "c":3} -> key value
pairs
- Set {1, "hello", 3} ->
collection of unique elements

Mathematical operations

Python has the following built-in mathematical operations.

Operator	Definition	Example
$x + y$	sum	$9 + 4 = 13$
$x - y$	subtraction	$9 - 4 = 5$
$x * y$	multiplication	$9 * 4 = 36$
x / y	division	$9 / 4 = 2.25$
$x // y$	integer division	$9 // 4 = 2$
$x \% y$	remainder (or modulo)	$9 \% 4 = 1$
$x ** y$	power	$9 ** 4 = 6561$

Mathematical operators

Two particular mathematical operators can have different meanings depending on the context:

1. Sum (+):

Except for numerical values, the sum operation means appending an element to another
-> appending usually requires two elements of same data types!

- ‘hello’ + ‘ ’ + ‘world’: ‘hello world’
- [1, 2, 3] + [4, 5]: [1, 2, 3, 4, 5]

2. Multiplication (*):

This operation **always** has to be applied with an **integer** (number).

- ‘hello’ * 3: ‘hello|hello|hello’
- [1, 2, 3] * 3: [1, 2, 3, 1, 2, 3, 1, 2, 3]

Finally, the two operations can be combined to:

- ‘hello’ * 3 + ‘hello’: ‘hello|hello|hello|hello’

Boolean operations

Two boolean values can be combined differently:

and

	True	False
True	True	False
False	False	False

or

	True	False
True	True	True
False	True	False

xor (^)

	True	False
True	False	True
False	True	False

True and False == False

True or False == True

True ^ False == False

Any boolean operation can be inverted with the keyword: **not**

True and not False == True

not True or False == False

not (True ^ False) == False

Course alignment

- Experiment with python scripts, consoles and basic math operations
 - > setup a new project
 - > set the interpreter to the venv
 - > write your python instructions with basic math operations, variable assignments and print commands
- Course book: data types, variables, comments
- Online course (tutorials.eu):
 - Section 2: Python basics
 - Section 3: Python basics part 2
 - Section 4: Control structures

School of Computer Science and Information Technology

Research

Ramón Christen

Research Associate Doctoral Student

Phone direct +41 41 757 68 96

ramon.christen@hslu.ch

HSLU T&A, Competence Center Thermal Energy Storage

Research

Andreas Melillo

Lecturer

Phone direct +41 41 349 35 91

andreas.melillo@hslu.ch