UNIVERSITÄT
ZU KÖLN

SPRACHVERARBEITUNG: ÜBUNG

SoSe 2024

Janis Pagel

Institut für Digital Humanities

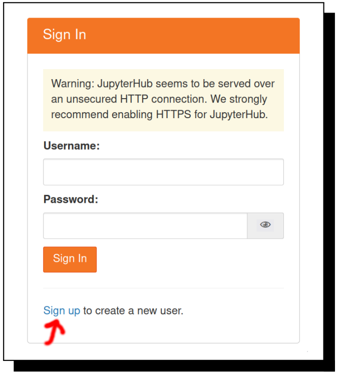
2024-04-09

01

JUPYTER

Jupyter Login

- `http://compute.spinfo.uni-koeln.de`
  - This is only reachable from the University of Cologne Network
  - `https://rrzk.uni-koeln.de/internetzugang-web/netzzugang/vpn` for VPN access



Sign In

Warning: JupyterHub seems to be served over an unsecured HTTP connection. We strongly recommend enabling HTTPS for JupyterHub.

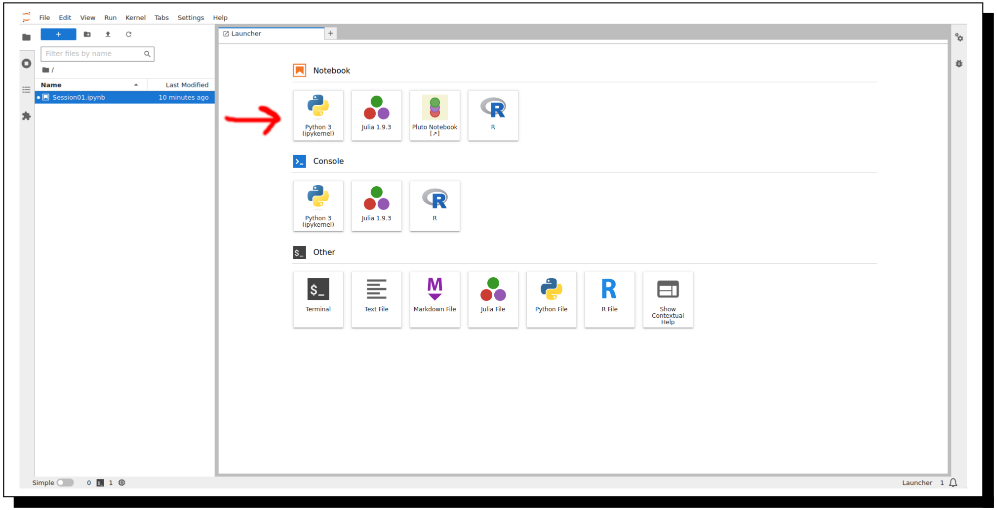
Username:

Password:

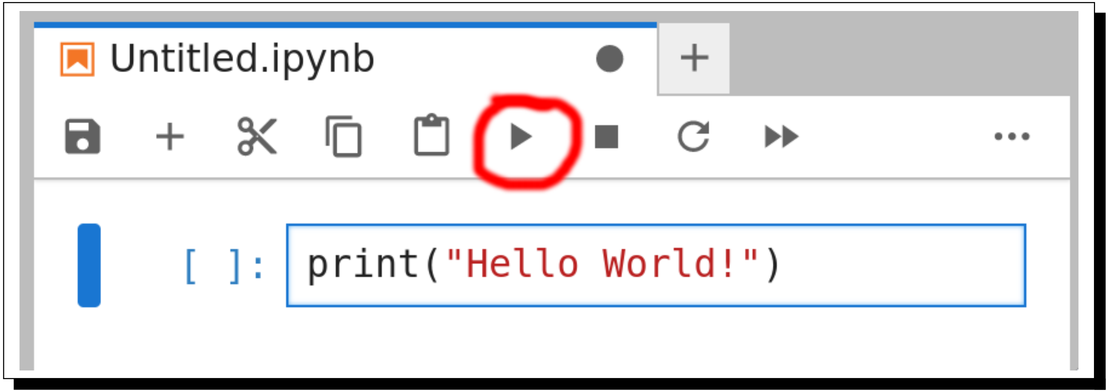
Sign up to create a new user.

- Sign up with a new username and password
- Go back to "Sign In" and sign in with your chosen username and password

Start Notebook

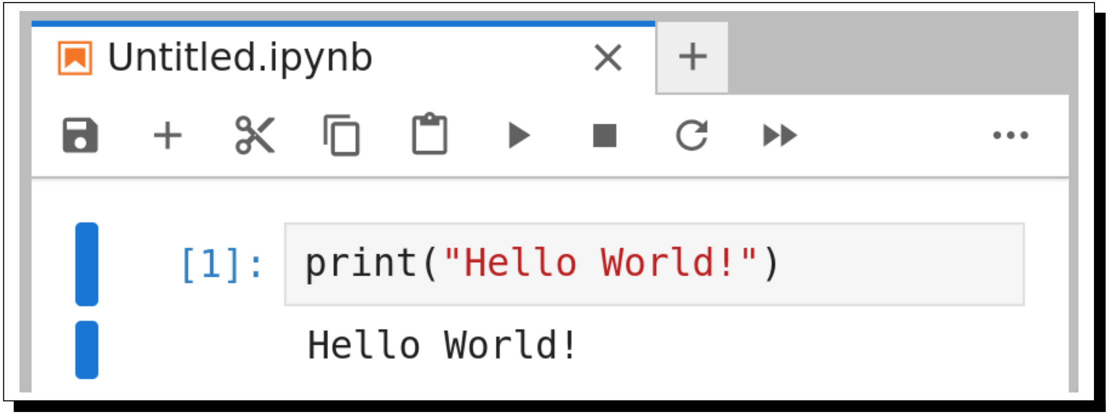


Run Code



```
[ ]: print("Hello World!")
```

Result of Running Code Cell



```
[1]: print("Hello World!")
```

```
Hello World!
```

02

PYTHON CRASH COURSE

Motivation

- Not a full-fledged Python course
- Enough knowledge to be able to use Python in the *Übung* and solve the exercises
- Assuming existing knowledge of Java
- If you do not know Java or any other programming language, it is still possible to attend the Übung, but it will be much harder

First Python Script

```
print("Hello world!")

> Hello World!
```

- Python is a script language, so the compilation and run processes are not separated, but done in one step by the Python interpreter
- Python does not need any class declaration to run (but you can use classes in Python if you wish)
- Some goals of Python: being nice to read, writing concise code
- `https://peps.python.org/pep-0008/`

Python Data Types I

- String

```
x = "banana"
print(x)
y = 'banana'
print(y)
print(x[0])
print(x[0:3]) # Slicing
print(x[0:1])
print(x[2:]) # Not giving the beginning/end of a slice goes until the end of the string
print(x[-2]) # Negative indices count from the end
z = "one"
print(z + " " + y) # Strings can be concatenated

> banana
> banana
> b
> ban
> b
> nana
> n
> one banana
```

- You can imaging the indices in Python being between characters for the purpose of slicing: ${}_0b_1a_2n_3a_4n_5a_6$

Python Data Types II

- List

```
empty_list = []
print(empty_list)
x = [1, 2, 3, 4]
y = ["banana", "apple", "coconut"]
print(x)
print(y)
print(y[0])
print(y[1:3])
empty_list.append("mango") # Items can be added to list via append method
print(empty_list)
print(y + empty_list) # Lists can be concatenated
y[0] = "orange" # List items can be changed
print(y)

> []
> [1, 2, 3, 4]
> ['banana', 'apple', 'coconut']
> banana
> ['apple', 'coconut']
> ['mango']
> ['banana', 'apple', 'coconut', 'mango']
> ['orange', 'apple', 'coconut']
```

Python Data Types III

- Dictionary (like *HashMap* in Java)

```
empty_dict = {}
print(empty_dict)
d = {"banana": "yellow", "apple": "red", "coconut": "brown"}
print(d)
print(d["coconut"])
d["cherry"] = "red"
print(d)
d["apple"] = "green"
print(d)
d2 = {"banana": ["yellow", "brown"], "apple": ["red", "green"]}
print(d2["banana"])
d3 = {"banana": {"color": "yellow", "sweet": True}, "coconut": {"color": "brown", "sweet": False}}
print(d3["coconut"]["sweet"])

> {}
> {'banana': 'yellow', 'apple': 'red', 'coconut': 'brown'}
> brown
> {'banana': 'yellow', 'apple': 'red', 'coconut': 'brown', 'cherry': 'red'}
> {'banana': 'yellow', 'apple': 'green', 'coconut': 'brown', 'cherry': 'red'}
> ['yellow', 'brown']
> False
```

If Statements I

```
x = ["banana", "apple", "coconut"]
if x[0][-1] == "a":
    print(f"The final letter of '{x[0]}' is 'a'")
else:
    print(f"The final letter of '{x[0]}' is not 'a'")

> The final letter of 'banana' is 'a'
```

```
x = ["banana", "apple", "coconut"]
if x[2][-1] == "a":
    print(f"The final letter of '{x[2]}' is 'a'")
elif x[2][0] == "c":
    print(f"The first letter of '{x[2]}' is 'c'")
else:
    print(f"The final letter of '{x[2]}' is not 'a' and the first letter is not 'c'")

> The first letter of 'coconut' is 'c'
```

- Python does not use curly brackets {} to group if statements and loops, but indentations
- Conventionally, one indentation should be a single tab or four spaces (spaces are preferred)
- The condition of the if statement is terminated via a colon :

If Statements II

```
x = ["banana", "apple", "coconut"]
if x[1][-1] == "a":
    print(f"The final letter of '{x[1]}' is 'a'")
elif x[1][0] == "c":
    print(f"The first letter of '{x[1]}' is 'c'")
else:
    print(f"The final letter of '{x[1]}' is not 'a' and the first letter is not 'c'")

> The final letter of 'apple' is not 'a' and the first letter is not 'c'
```

Python Loops I

- For Loop

```
for i in [1,2,3,4]:
    print(i)
for i in range(1,5):
    print(i)
# enumerate() creates a generator to iterate over list items plus an index
for i, fruit in enumerate(["banana", "apple", "coconut"]):
    print(i, fruit)
# zip() creates a generator to iterate over two lists in parallel
for item1, item2 in zip(["banana", "apple", "coconut"], ["yellow", "red", "brown"]):
    print(item1, item2)

> 1
> 2
> 3
> 4

> 1
> 2
> 3
> 4

> 0 banana
> 1 apple
> 2 coconut
```

Python Loops II

```
> banana yellow
> apple red
> coconut brown
```

Python Loops III

- While Loop

```
i = 1
while i <= 4:
    print(i)
    i+=1
> 1
> 2
> 3
> 4
```

Functions

```
def split_words(text):
    # The split() method operates on strings and outputs a list with items coming from the operation when the
                                                           string is split at the separator given to the
                                                           function
    return text.split(" ")
print(split_words("I want to split this text into a list containing its words"))
def split_lines(text):
    return text.split("\n")
# Three quotation marks around strings allows for multi-line strings
print(split_lines("""This text contains multiple lines.
I want to split it into a list containing one line per item."""))
def count_word_length(words):
    for word in words:
        print(len(word))
count_word_length(split_words("Count the word length of this text"))

> ['I', 'want', 'to', 'split', 'this', 'text', 'into', 'a', 'list', 'containing', 'its', 'words']
> ['This text contains multiple lines.', 'I want to split it into a list containing one line per item.']

> 5
> 3
> 4
> 6
> 2
> 4
> 4
```

- Input and output of functions are not typed in Python, you need to keep track of the type of a variable
- If a function does not have a return value, it does not need to be declared as *void*

03

OUTLOOK ON NEXT WEEK

Outlook

- Classes and Methods?
- Reading and writing files
- Useful libraries
    - pandas (Data Engineering)
    - seaborn (Plotting)

04

FURTHER MATERIAL ON PYTHON

Python Resources

- Install Python: `https://www.python.org/downloads/`
- Popular Python Development Environments
  - IDLE (built into Python): `https://docs.python.org/3/library/idle.html`, `https://www.python-lernen.de/python-idle.htm`
  - PyCharm: `https://www.jetbrains.com/pycharm`
  - spyder: `https://www.spyder-ide.org/`
  - VSCode: `https://code.visualstudio.com/`
- Python Tutorials
  - `https://docs.python.org/3/tutorial/index.html`
  - `https://python.land/python-tutorial`

UNIVERSITY
OF COLOGNE

Janis Pagel
Institut für Digital Humanities

eMail	janis.pagel@uni-koeln.de
Website	https://janispagel.de

Foto: Gregor Hübl