

Module 61-12: Option GIS-Python

Introduction to Python

hes.
so
business.

Jean-Paul Calbimonte

School of Management

Bachelor of Science HES-SO (BSc) in Business
Information Technology



> GIS for developers

Why Python for geodata?



- **Free:** no added costs for licensing
- **For coders:** fully programmable geodata manipulation
- **Modular:** libraries adapted to different use-cases
- **Efficiency:** optimized for Big Data analytics
- **Extensibility:** possibility to extend or reuse multiple libraries
- **Flexibility:** options for lots of formats/standards/approaches
- **Open Source:** code reuse/reproducibility/open science
- **Integration:** supported by other tools as QGIS/ArcGIS etc.

> How do we run python?

Different options

In this course:



- Online environment: Jupyter Python in renkulab.io
- Jupyter Python in Local installation

> renkulab.io

Log in with your hevs edu-ID account



The image shows the RenkuLab login interface. The background is a dark blue field with a pattern of small, colorful dots (pink, green, and yellow) arranged in a way that suggests a network or data points. In the top left corner, the 'renku' logo is visible in a light green, pixelated font. The main heading 'RenkuLab' is in a large, white, sans-serif font. Below it, a message in green and pink text reads: 'We have been improving RenkuLab. Welcome to the new version!'. At the bottom left, there are three buttons: 'Feedback' (with a speech bubble icon), 'Learn more' (with a magnifying glass icon), and 'Login' (with a person icon). On the right side, there is a white box titled 'SWITCH edu-ID' with a 'Help' link and a language dropdown set to 'EN'. Inside this box, the text 'Log in to: renkulab.io' is at the top. Below it, a 'Service description' section states: 'This is a public beta instance of the Renku platform for reproducible data science operated by the Swiss Data Science Center.' The login form includes an 'E-mail' field with the letter 'j' entered, and a 'Password' field with the placeholder 'Enter your password' and an eye icon for toggling visibility. There are 'Create account' and 'Login' buttons. Below the password field, there are links for 'Forgot password?' and 'Options for personal data protection'. At the bottom right of the login box is the 'SWITCH' logo. At the very bottom of the page, there is a footer with links: 'About / Terms of Use / Legal Notice / Imprint'.

renku

RenkuLab

We have been improving RenkuLab.
Welcome to the new version!

[Feedback](#) [Learn more](#) [Login](#)

SWITCH edu-ID

Help EN

Log in to: renkulab.io

Service description:
This is a public beta instance of the Renku platform for reproducible data science operated by the Swiss Data Science Center.

SWITCH edu-ID

E-mail:

Password: [Eye icon](#)

[Create account](#) [Login](#)

[Forgot password?](#)
[Options for personal data protection](#)

SWITCH

[About](#) / [Terms of Use](#) / [Legal Notice](#) / [Imprint](#)

> renkulab.io

Got to this project and fork it:

<https://renkulab.io/projects/jean-paul.calbimonte/gis-python/>

The screenshot shows the RenkuLab interface for a project named "GIS Python". At the top, the Renku logo is visible. The project title "GIS Python" is prominently displayed, along with a "Start" button. Below the title, it indicates the project is "Public" and lists the creator as "Jean-Paul Calbimonte Pérez". A description states: "GIS Python materials for the Bachelor in Business IT HES-SO Valais-Wallis". A row of icons (colon, sigma, pi, approx, and) is shown. Below this, there are icons for a globe and Python, with the text "Updated 3 months ago." A navigation bar includes links for Overview, Collaboration, Files, Datasets, Workflows, Sessions, and Settings. The "Fork" button is circled in red, with the word "fork" written in red above it. Other buttons include "View in GitLab" and a dropdown menu. A sidebar on the left contains links for General, Stats, Commits, and Status. The main content area shows a "README.md" file and the project title "GIS Python".

> renkulab.io

<https://renkulab.io/projects/jean-paul.calbimonte/gis-python/>

GIS Python ▶ Start ▼

Project Public

Jean-Paul Calbimonte Pérez
jean-paul.calbimonte/gis-python forked from [jean.paul.ik/gis-python](#)

GIS Python materials for the Bachelor in Business IT HES-SO Valais-Wallis

Updated 3 months ago.

Overview Collaboration Files Datasets Workflows **Sessions** Settings

❖ Back to jean-paul.calbimonte/gis-python

Sessions

No currently running sessions.

+ New session

Go to Sessions

Start a new session and open
it when it is ready

Default Environment
/lab /rstudio

Number of CPUs
0.25 0.5 1 2

Amount of Storage
1G 4G 16G 64G

☐ Automatically fetch LFS data

Environment Variables (Optional)
Add Variable

Amount of Memory
1G 2G 4G 8G

Number of GPUs **0**

Start session ▼

> renkulab.io

The screenshot shows the renkulab.io web interface. On the left is a file explorer with a sidebar containing icons for file operations. The main area of the file explorer shows a list of files and notebooks under the path '/ notebooks / 01-PythonIntro /'. The files are listed in a table with columns 'Name' and 'Last Modified'. The first file, '01-datatype...', is highlighted with a red arrow pointing to the 'Notebook' button in the main area of the interface. The main area is titled 'notebooks/01-PythonIntro' and contains a 'Notebook' button with a Python logo, a 'Console' button with a terminal icon, and an 'Other' section with buttons for 'Terminal', 'Text File', 'Markdown File', and 'Show Contextual Help'.

Name	Last Modified
01-datatype...	4 minutes ago
02-lists.ipynb	4 minutes ago
03-loops.ipynb	4 minutes ago
04-conditio...	4 minutes ago
05-function...	4 minutes ago
06-classes.i...	4 minutes ago
07-assertion...	4 minutes ago
08-files.ipynb	4 minutes ago
books.py	4 minutes ago
cantons.txt	4 minutes ago
exercise.ipynb	4 minutes ago
mats.py	4 minutes ago
test.txt	4 minutes ago

Ready to open the notebooks

> Running Jupyter

Basics of Python

> Basic operators

1 + 1

2

3 * 4.5

13.5

2 - 3.5

-1.5

10 / 3

3.3333333333333335

3 ** 2

9

basic arithmetic
operations

> Math functions

```
import math
```

```
math.sqrt(81)
```

```
9.0
```

```
math.pi
```

```
3.141592653589793
```

```
math.sin(math.pi/2)
```

```
1.0
```

import library

imported function

> Variables & Basic types

```
a=1
b=2.4      int
a+b        float
3.4
```

```
st='we like GIS'  str
print(st)
we like GIS
```

```
b=st
print(b)
we like GIS
```

```
print('This is number',a,'followed by this string:',b)
This is number 1 followed by this string: we like GIS
```

```
a==b      bool
False
a==1
True
```

> Basic types

```
type (a)
```

```
int
```

```
type (st)
```

```
str
```

```
type (3.42)
```

```
float
```

```
type (a==2)
```

```
bool
```

int
str
float
bool



> Lists

```
[3, 4, 6, 2, 1]
```

unidimensional

```
[3, 4, 6, 2, 1]
```

```
[[2, 3], [5, 6], [4, 3]]
```

multidimensional

```
[[2, 3], [5, 6], [4, 3]]
```

```
list=[3, 4.5, 6, 2, 2.1]
```

```
len(list)
```

```
5
```

```
list[2]
```

```
6
```

```
list[-1]
```

```
2.1
```

indices

```
type(list[1])
```

```
float
```

types

> Lists

```
list=[3,4.5,6]
```

```
del list[2]
```

```
print(list)
```

```
[3, 4.5]
```

remove item

```
list.append(44)
```

```
print(list)
```

```
[3, 4.5, 44]
```

append item

```
list.reverse()
```

```
print(list)
```

```
[44, 4.5, 3]
```

reverse list

```
list.sort()
```

```
print(list)
```

```
[3, 4.5, 44]
```

sort list

```
list[0]=55
```

```
print(list)
```

```
[55, 4.5, 44]
```

replace item

> Lists & Loops

```
string='strange'  
len(string)  
7
```

```
print(string[2])  
r
```

```
for ch in string:  
    print(ch)
```

```
s  
t  
r  
a  
n  
g  
e
```

iterate over chars

```
count=0  
for ch in string:  
    print(ch,count)  
    count=count+1  
print(count)  
s 0  
t 1  
r 2  
a 3  
n 4  
g 5  
e 6  
7
```

> Loops

```
range(4)  
range(0, 4)
```

```
for i in range(4):  
    print(i)
```

```
0  
1  
2  
3
```

iterate range

```
for i in range(2, 14, 3):  
    print(i)
```

```
2  
5  
8  
11
```

range with step

```
name='Aladdin'  
for i in range(len(name)):  
    print(name[i])
```

```
A  
l  
a  
d  
d  
i  
n
```

> Conditions

```
speed=80
```

```
if speed > 100:
```

```
    print('too fast')
```

```
elif speed > 80 and speed <=100 :
```

```
    print('speed ok')
```

```
else:
```

```
    print('too slow')
```

```
too slow
```

← if

← elif

← else

> Conditions

```
exam1 = 3.5  
exam2 = 'B'  
  
if (exam1 >=4 and exam2 == 'A') :  
    print('grades are great')  
elif exam1 < 4 and exam2 == 'B':  
    print('grades are poor')  
else:  
    print('grades are mixed')
```

grades are poor

> Functions

```
def calculateArea(length, height):  
    return length*height
```

```
calculateArea(5, 4)  
20
```

```
calculateArea('a', 4)  
'aaaa'
```

> Functions

```
def countLetters(strings, letter):  
    count=0  
    for str in strings:  
        count=count+str.count(letter)  
    return count
```

```
countLetters(['day', 'pasta', 'lasagna'], 'a')  
6
```

> Classes

```
class Vehicle(object):  
    wheels=0  
  
    def __init__(self,wheels,maxSpeed=0):           'constructor'  
        self.wheels=wheels  
        self.maxSpeed=maxSpeed  
  
    def fasterThan(self,otherVehicle):              method  
        return self.maxSpeed > otherVehicle.maxSpeed  
  
v1=Vehicle(2)                                       instantiation  
print(v1.maxSpeed)  
0
```

> Classes

```
class Bike(Vehicle):  
    def __init__(self, maxSpeed=0):  
        self.wheels=2  
        self.maxSpeed=maxSpeed
```

```
b1=Bike(30)  
b1.wheels  
b1.maxSpeed  
30
```

```
b2=Vehicle(4, 80)  
b1.fasterThan(b2)  
False
```

> Error

```
def calculateArea(length,height):  
    return length*height
```

```
calculateArea('a','dsd')
```

```
-----  
-----  
TypeError Traceback (most recent call last)  
<ipython-input-2-2ccc00b376b5> in <module>  
----> 1 calculateArea('a','dsd')
```

```
<ipython-input-1-bf7816c63e4e> in calculateArea(length,  
height)
```

```
      1 def calculateArea(length,height):  
----> 2 return length*height
```

```
TypeError: can't multiply sequence by non-int of type 'str'
```

> Assertions

```
def calculateArea(length,height):  
    assert length > 0 , 'length must be positive'  
    assert height > 0 , 'height must be positive'  
    assert type(length) == float  
calculateArea(-3.0,2)
```

```
(-----  
(AssertionError Traceback (most recent call last)  
<ipython-input-15-f962da30df2f> in <module>  
----> 1 calculateArea(-3.0,2)  
  
<ipython-input-12-f5dade16329c> in calculateArea(length, height)  
      3 #assert type(height) == float  
      4  
----> 5 assert length > 0 , 'length must be positive'  
      6 return length*height  
AssertionError: length must be positive
```

> Questions?

Thank you for your attention.

hes.
so
you.

School of Management
Route de la Plaine 2
3960 Sierre

hevs.ch/heg



swissuniversities

