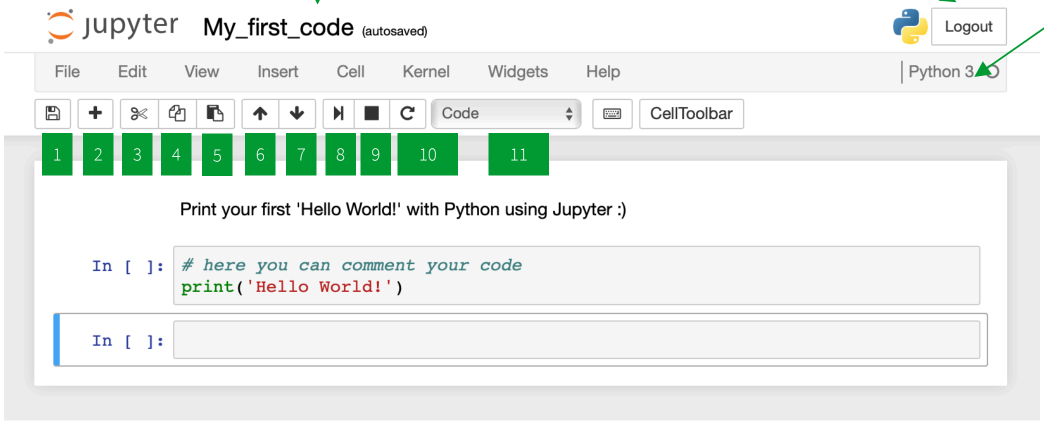
Create your first Jupyter notebook

1. Instructor will provide you the **address** of the **VM** -> type it in your browser.
2. Go to “Start” and type ‘cmd’ in the search bar -> command terminal will open.
3. Type *‘jupyter notebook’* and press “Enter” -> jupyter notebook will open…

Cheat sheet Python and Jupyter



Print your first 'Hello World!' with Python using Jupyter :)

```
In [ ]: # here you can comment your code
        print('Hello World!')

In [ ]:
```

Title

Log out from notebook server

Current kernel

1 Save and checkpoint

2 Insert cell below

3 Cut cell

4 Copy cell(s)

5 Paste cell(s) below

6 Move cell up

7 Move cell down

8 Run current cell

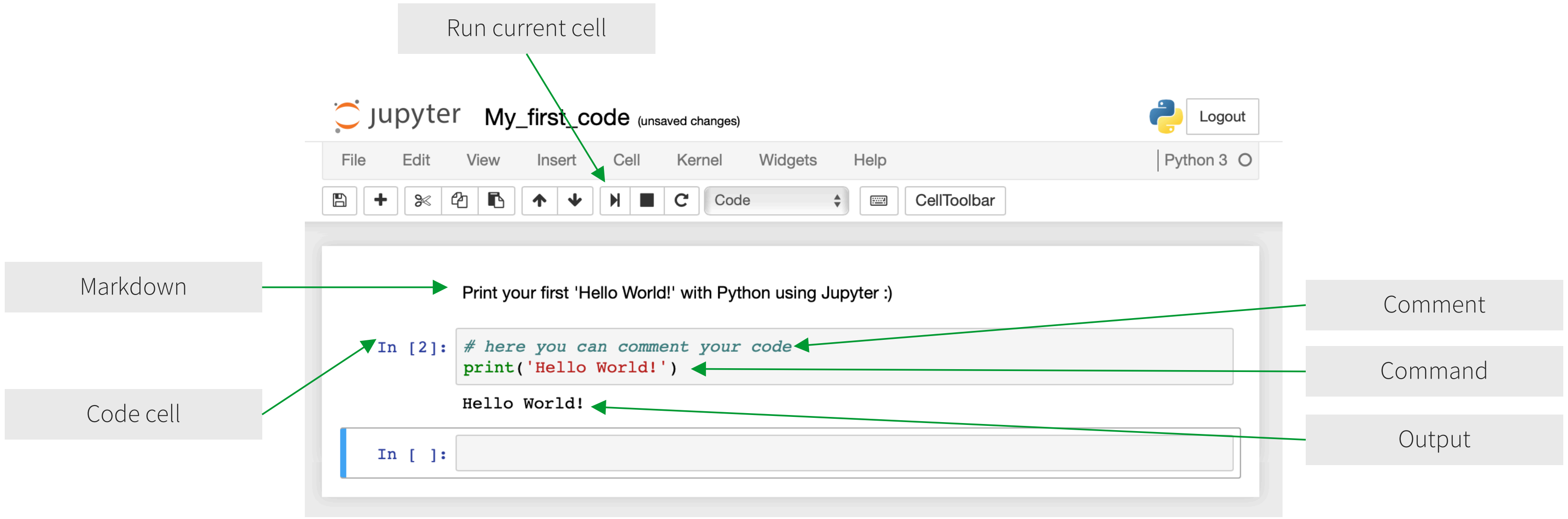
9 Interrupt kernel

10 Restart kernel

11 Display characteristics

More information: https://datacamp-community-prod.s3.amazonaws.com/48093c40-5303-45f4-bbf9-0c96c0133c40

Cheat sheet Python and Jupyter



Print your first 'Hello World!' with Python using Jupyter :)

```
# here you can comment your code
print('Hello World!')
```

```
Hello World!
```

Python Basics

Variable Assignment

```
x = 6
x
```

```
6
```

Calculations

```
x+2 # Sum of two variables
```

```
8
```

```
x-2 # Substraction of two variables
```

```
4
```

```
x*2 # Mulsplication of two variables
```

```
12
```

```
x**2 # Exponentiation of a variable x^2
```

```
36
```

```
x%2 # Remainder of a variable - modulo
```

```
0
```

```
x/4 # Division of a variable
```

```
1.5
```

Types and Type conversion

```
str(3.0) # Variables to strings
```

```
'3.0'
```

```
int(5.3) # Variables to integers
```

```
5
```

```
float(5) # Variables to floats
```

```
5.0
```

```
bool() # True/False -  Variables to booleans
```

Import Libraries

```
# Import libraries
import pandas as pd
import numpy as np
# Selective import
from math import pi
```

Asking for Help

```
help(str)
```

```
Help on class str in module builtins:

class str(object)
```

Lists

```
a = 'is'
b = 'nice'
my_list = ['my', 'list', a, b]
my_list_2 = [[4,5,6,7],[3,4,5,6]]

my_list[1] # select item at index 1 -> 'list'
my_list[-2] # select 2nd last item -> 'is'
```

Exercise

Create the list from above and find out what the following operations do..

```
my_list[1:3]
my_list[1:]
my_list[:3]
my_list[:]
my_list_2[1][0]
my_list_2[1][:2]
my_list = my_list + my_list
my_list.index(a)
my_list.count(a)
my_list.append('!')
my_list.remove('!')
```

Python Packages APIs

Links

Python Standard Library API Reference
https://docs.python.org/3/library/

Pandas Library API Reference
http://pandas.pydata.org/pandas-docs/stable/reference/index.html

NumPy Library API Reference
https://docs.scipy.org/doc/numpy/reference/

Seaborn Library API Reference
https://seaborn.pydata.org/api.html#

Python Data Science

Import Data

Most of the time, you'll use either **NumPy** or **Pandas** to import and process your data.

```
import numpy as np
import pandas as pd
```

Plain Text Files

```
filename = 'huck_finn.txt'
file = open(filename, mode='r')   Open the file for reading
text = file.read()                Read a file's contents
print(file.closed)                Check whether file is closed
file.close()                      Close file
print(text)
```

Table Data: Flat Files

...with NumPy

Files with one data type

```
filename = 'mnist.txt'
data = np.loadtxt(filename,
                  delimiter=',',   String used to separate values
                  skiprows=2,      Skip the first 2 lines
                  usecols=[0,2],   Read the 1st and 3rd column
                  dtype=str)       The type of the resulting array
```

Files with mixed data type

```
filename = 'titanic.csv'
data = np.genfromtxt(filename,
                     delimiter=',',
                     names=True,
                     dtype=None)
```

```
data_array = np.recfromcsv(filename)
```

...with Pandas

```
filename = 'winequality-red.csv'
data = pd.read_csv(filename,
                   nrows=5,          Number of rows of file to read
                   header=None,      Row number to use as col names
                   sep='\t',         Delimiter to use
                   comment='#',      Character to split comments
                   na_values=[""])   String to recognize as NA/NaN
```

Excel Spreadsheets

```
file = 'urbanpop.xlsx'
data = pd.ExcelFile(file)
df_sheet_1 = data.parse('1960-1966',
                        skiprows=[0],
                        names=['Country',
                               'AAM: War(2002)'])
df_sheet_2 = data.parse(0,
                        parse_cols=[0],
                        names=['Country'])
```

Exploring your Data

NumPy Arrays

```
data_array.dtype
data_array.shape
len(data_array)
```

pandas DataFrames

```
df.head()                          Return first DataFrame rows
df.head(10)                        Return first 10 DataFrame rows
df.tail()                          Return last DataFrame rows
df.index                           Describe index
df.columns                         Describe DataFrame columns
df.info()                          Info on DataFrame
data_array = data.values           Convert a DataFrame to an NumPy array
```

Navigating your File System

os Library

```
import os
path = "usr/tmp"
wd = os.getcwd()                   Store the name of current directory in a string
os.listdir(wd)                     Output contents of the directory in a list
os.chdir(path)                     Change current working directory
os.rename("test_1.txt",            Rename a file
          "test_2.txt")
os.remove("test_1.txt")            Delete an existing file
os.mkdir("newdir")                 Create a new directory
```

Python Pandas

Import Data

The Pandas library is built on NumPy and provides easy-to-use data structures and data analysis tools for the python programming language.

Pandas Data Structure

Series

A one-dimensional labeled array capable of holding any data type

Index

a	5
b	-3
c	2
d	7

```
s = pd.Series([5, -3, 2, 7], index=['a','b','c','d'])
```

DataFrame

A two-dimensional labeled data structure with columns of potentially different

	Title	Author	Price
0	Emma	Austen	€ 4,30
1	Dracula	Stoker	€ 10,00
2	Ivanhoe	Scott	€ 20,00

```
data = {'Title': ['Emma','Dracula','Ivanhoe'],
        'Author': ['Austen','Stoker','Scott'],
        'Price': [4.3,10,20]}
df = pd.DataFrame(data,
                  columns=['Title','Author','Price'])
```

Selection

Selecting, Boolean Indexing & Setting

By Position

```
df.iloc[[0],[0]]
```

Select single value by row & column

	Title
0	Emma

By Label

```
df.loc[[0], ['Author']]
```

Select single value by row & column labels

	Author
0	Austen

By Label/Position

```
df.iloc[2]
```

Select single row of subset of rows

```
Title      Ivanhoe
Author       Scott
Price           20
Name: 2, dtype: object
```

```
df.loc[:,'Author']
```

Select single column of subset of columns

```
0     Austen
1     Stoker
2      Scott
Name: Author, dtype: object
```

```
df.Title
```

Select single column of subset of columns

```
0        Emma
1     Dracula
2     Ivanhoe
Name: Title, dtype: object
```

Boolean Indexing

```
s[~(s > 2)]
```

Series S where value is not > 2

```
b   -3
c    2
dtype: int64
```

```
s[(s < -1) | (s > 5 )]
```

Series S where value is < -1 or > 5

```
b   -3
d    7
dtype: int64
```

```
df[df.Price > 8]
```

Subset of df where Price is > 8

	Title	Author	Price
1	Dracula	Stoker	10.0
2	Ivanhoe	Scott	20.0

Setting

```
s['a'] = 6
```

Set index a of Series S to 6

Python Pandas & Seaborn

Dropping and Sort

`s.drop(['a', 'c'])`	Drop values from rows
`df.drop('Author',axis = 1)`	Drop values from columns
`df.sort_index()`	Sort by labels along an axis
`df.sort_values(by='Title')`	Sort by the values along an axis
`df.rank()`	Assign ranks to entries

Retrieving Series/DataFrame Information

Basic Information

`df.shape`	(rows, columns)
`df.index`	Describe index
`df.columns`	Describe DataFrame columns
`df.info()`	Info on DataFrame
`df.count()`	Number of non-NA values

Summary

`df.sum()`	Sum of values
`df.cumsum()`	Cumulative sum of values
`df.min() / df.max()`	Minimum/maximum values
`df.idxmin() / df.idmax()`	Minimum/maximum index value
`df.describe()`	Summary statistics
`df.mean()`	Mean of values
`df.median()`	Median of values

Handling Missing Data

```
df.dropna()
df.fillna(value)
```

Exploring Data

`df.isnull()`	Detect missing values for an array-like object.
`len(df)`	Number of rows in DataFrame
`df.Author.value_counts()`	Return a Series containing counts of unique values.

Group Data

`df.groupby(by="col")`	Return a GroupBy object, grouped by values in column named "col".
`df.groupby(level="ind")`	Return a GroupBy object, grouped by values in index named "ind".
`df.groupby(by="col").size()`	Size of each group
`df.groupby(by="col").agg(function)`	Aggregate group using function.

Transforming Data

`df.astype('category')`	Cast a pandas object to a specified data type (e.g. 'category').
`pd.get_dummies(df, columns=["Author"])`	Convert categorical variable into dummy variables.

Plotting with Pandas

`df.plot.hist()`	Histogram for each column.
`df.plot.scatter(x='w', y='h')`	Scatter chart using pairs of points

Plotting with Seaborn

Import

```
import matplotlib.pyplot as plt
import seaborn as sns
```

1. Data

Use your **pandas** *DataFrame* or built-in data sets offered by Seaborn.

2. Figure Aesthetics

```
sns.set()
sns.set_style("whitegrid")
sns.axes_style("whitegrid")
```

3. Plotting with Seaborn

```
sns.stripplot(x="species",y="petal_length",data="iris")
sns.barplot(x="sex",y="survived",hue="class",data=titanic)
sns.countplot(x="deck",data=titanic,palette="Greens_d")
sns.boxplot(x="alive",y="age",hue="adult_male",data=titanic)
```

4. Further Customizations

```
plt.title("A Title")
plt.ylabel("Survived")
plt.xlabel("Sex")
plt.ylim(0,100)
plt.xlim(0,10)
```

5. Show or Save Plot

```
plt.show()
plt.savefig("foo.png")
```