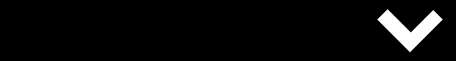
ITP 449
Lecture 13

Working with APIs



What is API

- Application Programming Interface
- A set of rules to allow an application to request data from another system without having direct access to their database
- Protocol rules set the requirements for requests
- Not all systems have API interfaces

API Request Example

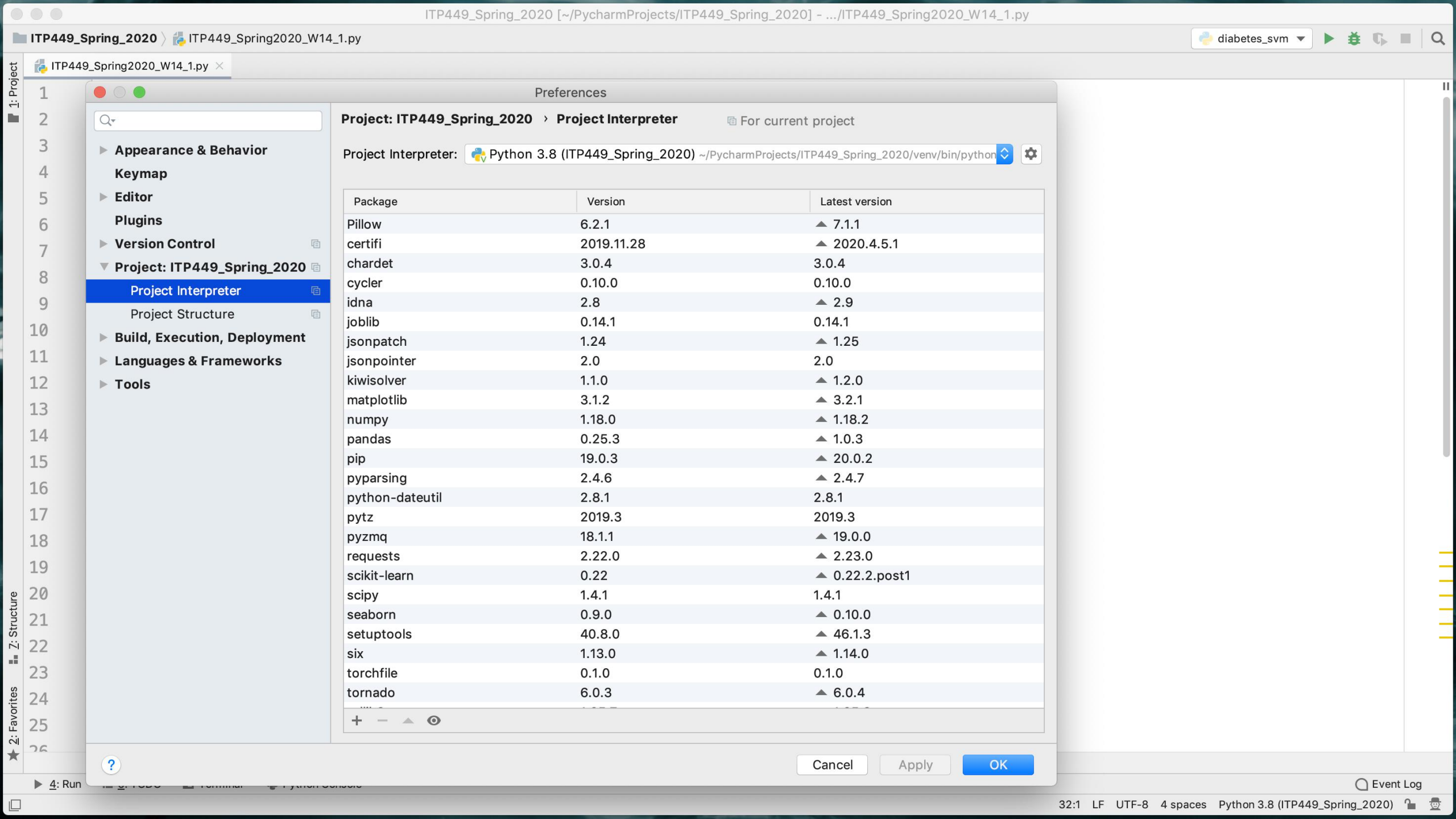
- Google Map
  - https://maps.googleapis.com/maps/api/timezone/json?location=39.6034810,-119.6822510&timestamp=1331161200&key=YOUR_API_KEY
- Census Data
  - https://api.census.gov/data/2010/dec/sf1?get=H001001,NAME&for=state:*&key=[user key]

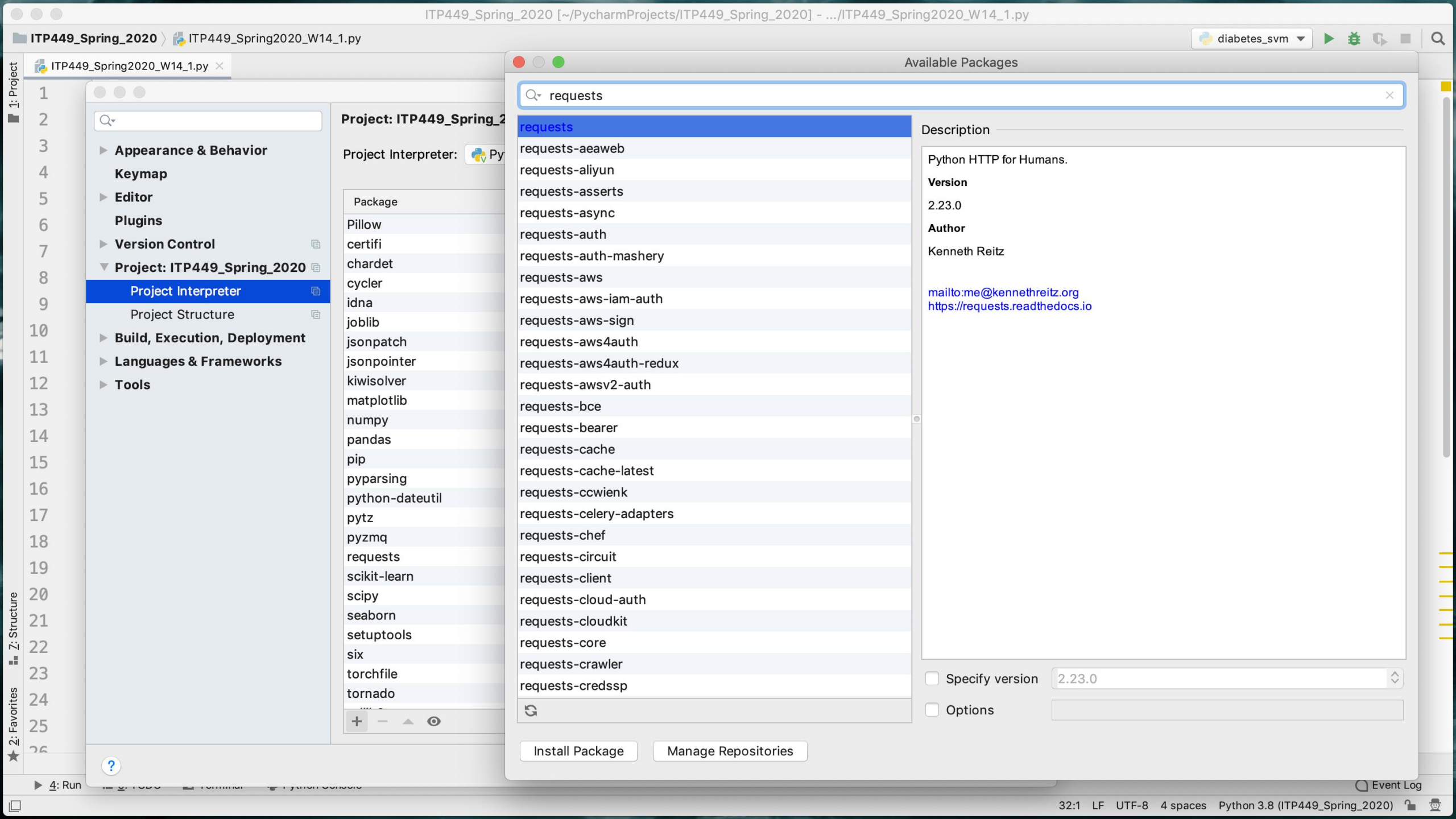
API Structure

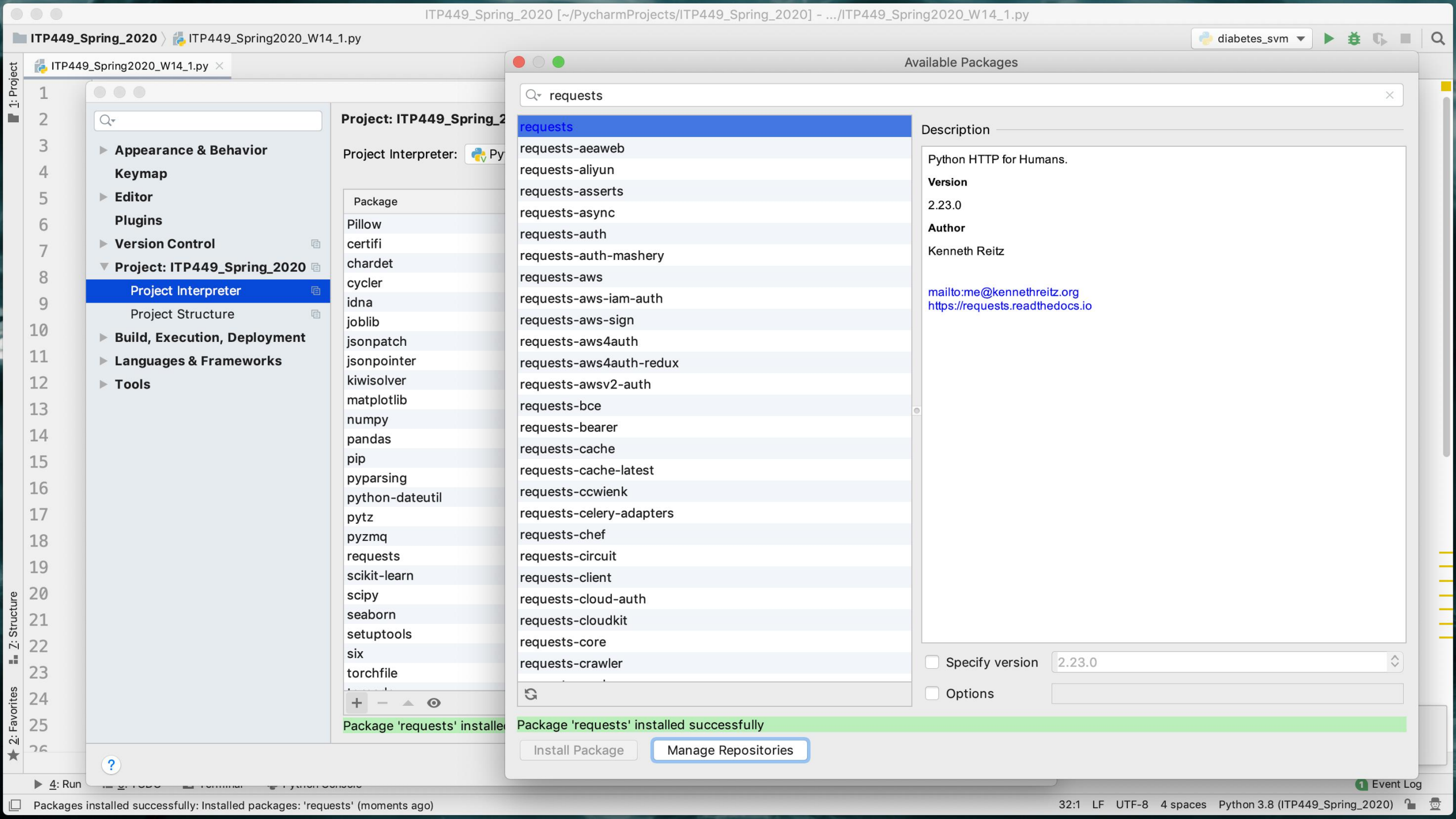
- https://api.census.gov/data/2010/dec/sf1?get=H001001,NAME&for=state:*&key=[user key]
- Base URL
  - *Host* https://api.census.gov/data
  - *Year* 2010
  - *Dataset* dec/sf1
- Parameters (after the ?)
  - Query string
  - *Get* a list of variables
  - *For* geography of interest
- Response from the server
- JSON format

JSON

- JSON (Java Script Object Notation) to describe data
  - Java Script Object Notation (JSON)
  - Curly Bracket ‘{ }’ – start/end
  - Square Bracket ‘[ ]’ – start/end an element or array
  - Comma ‘,’ – field separator
  - Colon ‘:’ – key value separator


ITP449_Spring_2020 [~/PycharmProjects/ITP449_Spring_2020] - .../ITP449_Spring2020_W14_1.py
ITP449_Spring_2020
ITP449_Spring2020_W14_1.py
diabetes_svm
ITP449_Spring2020_W14_1.py
Preferences
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Project: ITP449_Spring_2020
Project Interpreter
Project Structure
Build, Execution, Deployment
Languages & Frameworks
Tools
Project: ITP449_Spring_2020 › Project Interpreter
For current project
Project Interpreter: Python 3.8 (ITP449_Spring_2020) ~/PycharmProjects/ITP449_Spring_2020/venv/bin/python
Package
Version
Latest version
Pillow 6.2.1 7.1.1
certifi 2019.11.28 2020.4.5.1
chardet 3.0.4 3.0.4
cycler 0.10.0 0.10.0
idna 2.8 2.9
joblib 0.14.1 0.14.1
jsonpatch 1.24 1.25
jsonpointer 2.0 2.0
kiwisolver 1.1.0 1.2.0
matplotlib 3.1.2 3.2.1
numpy 1.18.0 1.18.2
pandas 0.25.3 1.0.3
pip 19.0.3 20.0.2
pyparsing 2.4.6 2.4.7
python-dateutil 2.8.1 2.8.1
pytz 2019.3 2019.3
pyzmq 18.1.1 19.0.0
requests 2.22.0 2.23.0
scikit-learn 0.22 0.22.2.post1
scipy 1.4.1 1.4.1
seaborn 0.9.0 0.10.0
setuptools 40.8.0 46.1.3
six 1.13.0 1.14.0
torchfile 0.1.0 0.1.0
tornado 6.0.3 6.0.4
Cancel
Apply
OK
1: Project
2: Favorites
7: Structure
4: Run
Event Log
32:1 LF UTF-8 4 spaces Python 3.8 (ITP449_Spring_2020)


ITP449_Spring_2020 [~/PycharmProjects/ITP449_Spring_2020] - .../ITP449_Spring2020_W14_1.py
ITP449_Spring_2020
ITP449_Spring2020_W14_1.py
diabetes_svm
ITP449_Spring2020_W14_1.py
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Project: ITP449_Spring_2020
Project Interpreter
Project Structure
Build, Execution, Deployment
Languages & Frameworks
Tools
Project: ITP449_Spring_2
Project Interpreter:
Package
Pillow
certifi
chardet
cycler
idna
joblib
jsonpatch
jsonpointer
kiwisolver
matplotlib
numpy
pandas
pip
pyparsing
python-dateutil
pytz
pyzmq
requests
scikit-learn
scipy
seaborn
setuptools
six
torchfile
tornado
Available Packages
requests
requests
requests-aeaweb
requests-aliyun
requests-asserts
requests-async
requests-auth
requests-auth-mashery
requests-aws
requests-aws-iam-auth
requests-aws-sign
requests-aws4auth
requests-aws4auth-redux
requests-awsv2-auth
requests-bce
requests-bearer
requests-cache
requests-cache-latest
requests-ccwienk
requests-celery-adapters
requests-chef
requests-circuit
requests-client
requests-cloud-auth
requests-cloudkit
requests-core
requests-crawler
requests-credssp
Description
Python HTTP for Humans.
Version
2.23.0
Author
Kenneth Reitz
mailto:me@kennethreitz.org
https://requests.readthedocs.io
Specify version
2.23.0
Options
Install Package
Manage Repositories
4: Run
Event Log
32:1
LF
UTF-8
4 spaces
Python 3.8 (ITP449_Spring_2020)


ITP449_Spring_2020 [~/PycharmProjects/ITP449_Spring_2020] - .../ITP449_Spring2020_W14_1.py
ITP449_Spring_2020
ITP449_Spring2020_W14_1.py
diabetes_svm
ITP449_Spring2020_W14_1.py
Appearance & Behavior
Keymap
Editor
Plugins
Version Control
Project: ITP449_Spring_2020
Project Interpreter
Project Structure
Build, Execution, Deployment
Languages & Frameworks
Tools
Project: ITP449_Spring_2
Project Interpreter:
Package
Pillow
certifi
chardet
cycler
idna
joblib
jsonpatch
jsonpointer
kiwisolver
matplotlib
numpy
pandas
pip
pyparsing
python-dateutil
pytz
pyzmq
requests
scikit-learn
scipy
seaborn
setuptools
six
torchfile
Package 'requests' installe
Available Packages
requests
requests
requests-aeaweb
requests-aliyun
requests-asserts
requests-async
requests-auth
requests-auth-mashery
requests-aws
requests-aws-iam-auth
requests-aws-sign
requests-aws4auth
requests-aws4auth-redux
requests-awsv2-auth
requests-bce
requests-bearer
requests-cache
requests-cache-latest
requests-ccwienk
requests-celery-adapters
requests-chef
requests-circuit
requests-client
requests-cloud-auth
requests-cloudkit
requests-core
requests-crawler
Description
Python HTTP for Humans.
Version
2.23.0
Author
Kenneth Reitz
mailto:me@kennethreitz.org
https://requests.readthedocs.io
Specify version
2.23.0
Options
Package 'requests' installed successfully
Install Package
Manage Repositories
4: Run
Event Log
Packages installed successfully: Installed packages: 'requests' (moments ago)
32:1
LF
UTF-8
4 spaces
Python 3.8 (ITP449_Spring_2020)

INTERNET ARCHIVE

Donate ♥

OPEN LIBRARY

Browse ▾ My Books ▾ More ▾ All ▾ Search Log In Sign Up

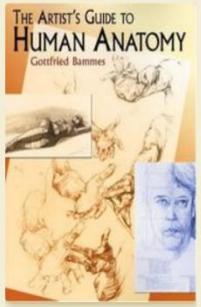
Together, let's build an Open Library for the World. Sponsor a Book

https://openlibrary.org/

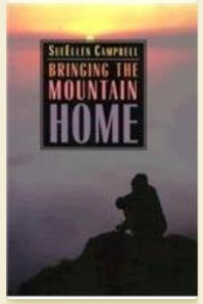
Browse by Subject

- Art — 45,803 Books
- Fantasy — 11,223 Books
- Biographies — 8,468 Books
- Science — 49,746 Books
- Recipes — 6,470 Books
- Romance — 17,122 Books

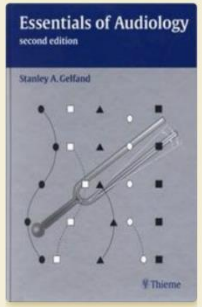
Books to Sponsor



Sponsor

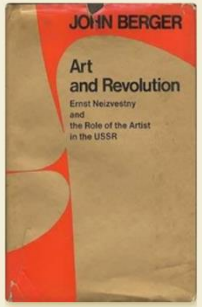


Sponsor

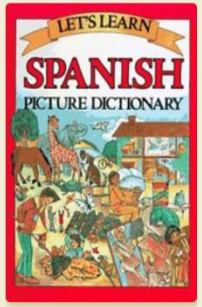


Sponsor

Sponsor

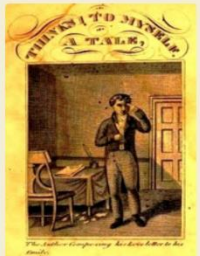


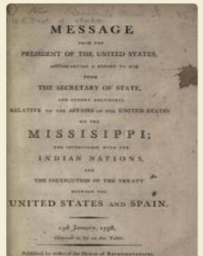
Sponsor

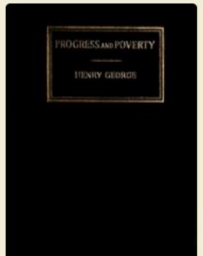


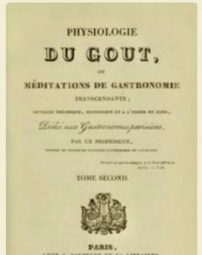
Sponsor

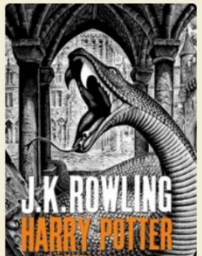
Classic Books

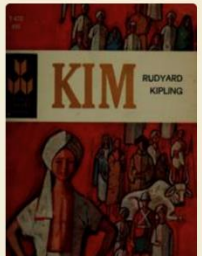












Developer Center

Last edited by Mek
September 29, 2018 | History

Edit

Developers | API | Data | Licensing

- Bugs | Source Code | OL Development

Hello! There are lots of ways you can work with the Open Library system, and the ol-tech mailing list is a great place to ask questions.

Open Library is powered by Infogami, a clean, simple wiki application framework built on web.py. Unlike other wikis, Infogami has the flexibility to handle different classes of data, including structured data. That makes it the perfect platform for Open Library.

Open Library also uses a text-to-HTML formatting language Markdown, created by John Gruber. We also use the handy WMD Markdown WYSIWYG editor.

Special thanks to Browserstack for cross browser compatibility testing.

Developer Notes & Blogs
- Current documentation pages
- Main older documentation page & API Documentation
- Ruby interface for the API, written by Jay Fajardo - Using covers
- Set up a development instance of Open Library
- Committing changes to github.com/openlibrary/openlibrary
- Writing a Bot for Open Library
- Using the Internet Archive BookReader
- Archive.org book URLs
- In-Library lending tech notes
- Github Issue Tracker, old issue tracker on launchpad
- Blogs by current and former devs: Gio, Raj, Anand

Data

- Bulk Download Open Library Bibliographic Data
- Archive.org Interfaces to publicly-accessible books and updates

Infogami

https://openlibrary.org/developers

Developers / API

Last edited by Charles Horn
October 19, 2018 | History

Edit

Developers | **API** | Data | Licensing

- Bugs | Source Code | OL Development

Open Library has developed a suite of APIs to help developers get up and running with our data. We encourage interested developers to join the ol-tech mailing list to stay up-to-date with the latest news, or dive in with our own development team at our bug tracker or our GitHub source code repository.

APIs

Open Library has a **RESTful API**, best used to link into Open Library data in JSON, YAML and RDF/XML. There's also an earlier JSON API, which is deprecated now. This is only retained for backward compatibility.

You can also return bibliographic data by simply adding an .rdf/.json/.yml extension to the end of any OL bibliographic identifier. There are also links on pages to RDF and JSON versions of Works, Editions and Authors.

Read APIs

- Books
- Covers
- Lists
- Read – *new!*
- Recent Changes
- Search
- Search inside
- Subjects

https://openlibrary.org/developers/api

Access to Records in Bulk

Our API should not be used for bulk download of Open Library data. If you want a dump of complete data, please read about your Bulk Download options, or email us at info@.

Subjects API

Last edited by Anand Chitipothu
October 12, 2010 | History

This API is experimental. Please be aware that this may change in future.

Get works of a subject

Sample Request:

```
GET /subjects/love.json
```

Sample Response:

```
{
    "key": "/subjects/love",
    "name": "love"
    "subject_type": "subject",
    "work_count": 4918,
    "works": [
        {
            "key": "/works/OL66534W",
            "title": "Pride and prejudice",
            "edition_count": 752,
            "authors": [
                {
                    "name": "Jane Austen",
                    "key": "/authors/OL21594A"
                }
            ],
            "has_fulltext": true,
            "ia": "mansfieldparknov03aust",
            ...
        },
        ...
    ]
}
```

https://openlibrary.org/dev/docs/api/subjects

All details of a subject

When query parameter `details=true` is passed, related subjects, prominent publishers, prolific authors and publishing_history are also included in the response.

Sample Request:

Try this

http://openlibrary.org/subjects/love.json

And this

http://openlibrary.org/subjects/love.json?published_in=1500-1600

JSON Parser Jason formatter

https://chrome.google.com/webstore/detail/json-formatter/bcjindcccaagfpapjjmafapmmgkkhgoa?hl=en

Then try this
http://openlibrary.org/subjects/love.json?published_in=1500-1600

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - apis.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 apis.py

apis.py

```
1  import requests
2
3  url = "http://openlibrary.org/subjects/love.json"
4  response = requests.get(url)
5
6  print(response.status_code)
7
8
```

Enter this URL in a browser and see what you get
http://openlibrary.org/subjects/love.json

Also, status code 200 means OK

What is status code 404?

Run: apis

```
"C:\Users\Nitin Kale\venv\Scripts\python.exe" "C:/Users/Nitin Kale/PycharmProjects/ITP449 Master/Lecture 13/apis.
200

Process finished with exit code 0
```

6: TODO 4: Run 5: Debug Terminal Python Console 2 Event Log

Packages installed successfully: Installed packages: 'requests' (7 minutes ago) 2:1 CRLF UTF-8 4 spaces Python 3.8 (venv)

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - apis.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 apis.py

apis

1: Project

7: Structure

apis.py

```
1  import requests
2  import json
3
4  url = "http://openlibrary.org/subjects/love.json"
5  response = requests.get(url)
6
7  print(response.status_code)
8
9  data = json.loads(response.text)
10
11 print(type(response.text))
12 print(type(data))
```

Run: apis

```
"C:\Users\Nitin Kale\venv\Scripts\python.exe" "C:/Users/Nitin Kale/PycharmProjects/ITP449 Master/Lecture 13/apis.
200
<class 'str'>
<class 'dict'>

Process finished with exit code 0
```

2: Favorites

6: TODO 4: Run 5: Debug Terminal Python Console 2 Event Log

PEP 8: W292 no newline at end of file 12:18 CRLF UTF-8 4 spaces Python 3.8 (venv)

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - apis.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 apis.py

apis

apis.py

```
5  response = requests.get(url)
6
7  print(response.status_code)
8
9  data = json.loads(response.text)
10
11 print(type(response.text))
12 print(type(data))
13
14 print(json.dumps(data, indent=5))
```

Run: apis

```
"C:\Users\Nitin Kale\venv\Scripts\python.exe" "C:/Users/Nitin Kale/PycharmProjects/ITP449 Master/Lecture 13/apis.
200
<class 'str'>
<class 'dict'>
{
     "subject_type": "subject",
     "name": "love",
     "key": "/subjects/love",
     "ebook_count": 11263,
     "works": [
          {
```

6: TODO 4: Run 5: Debug Terminal Python Console Event Log

Packages installed successfully: Installed packages: 'requests' (9 minutes ago) 12:18 CRLF UTF-8 4 spaces Python 3.8 (venv)

```
{
    "subject_type": "subject",
    "name": "love",
    "key": "/subjects/love",
    "ebook_count": 10350,
    "works": [
        {
            "printdisabled": true,
            "cover_id": 8256292,
            "ia_collection": [
                "toronto",
                "JaiGyan",
                "digitallibraryindia",
                "americana",
                "duke_libraries",
                "delawarecountydistrictlibrary",
                "librivoxaudio",
                "gutenberg",
                "audio_bookspoetry",
                "china",
                "trent_university",
                "inlibrary",
                "printdisabled",
                "internetarchivebooks",
                "cornell",
                "librarygenesis",
                "opensource",
                "robarts",
                "europeanlibraries"
            ],
            "has_fulltext": true,
            "edition_count": 475,
            "checked_out": false,
            "title": "Wuthering Heights",
            "public_scan": true,
            "cover_edition_key": "OL26500373M",
            "lending_edition": "OL11072346M",
            "first_publish_year": null,
            "key": "/works/OL15035771W",
            "authors": [
                {
                    "name": "Emily Bront\u00eb",
                    "key": "/authors/OL4327048A"
                }
            ],
            "ia": "wuthering_heights_0801_librivox",
            "lending_identifier": "wutheringhgts00emil",
            "availability": {
                "status": "open",
                "openlibrary_work": null,
                "isbn": null,
                "num_waitlist": null,
                "last_loan_date": null,
                "openlibrary_edition": null,
                "oclc": null,
                "collection": "librivoxaudio,audio_bookspoetry,fa
                "last_waitlist_date": null,
                "identifier": "wuthering_heights_0801_librivox"
            },
            "subject": [
                "Heathcliff (Fictitious character)",
                "Man-woman relationships",
                "love",
                "Accessible book",
                "Fiction",
                "England in fiction",
                "Literatuurkritiek",
                "Historical Fiction",
                "Orphans in fiction",
                "Foundlings",
                "Wuthering Heights (Bront\u00eb, Emily)",
                "death",
                "Reading Level-Grade
                "Reading Level-Grade 7",
                "Reading Level-Grade 9",
                "Reading Level-Grade 8",
                "Man-woman relationships in fiction",
                "romance",
                "French fiction",
                "Landscape in literature in fiction",
                "Country life in fiction",
                "Classic Literature",
                "foundlings in fiction",
                "Love in fiction",
                "Open Library Staff Picks",
                "Landscape in literature",
                "revenge",
                "Rejection (Psychology)",
                "Drama",
                "Country life",
                "Triangles (Interpersonal relations)",
                "Reading Level-Grade 10",
                "Manners and customs",
                "Rural families",
                "Rural families in fiction",
                "Social life and customs",
                "tragedy in fiction",
                "orphans",
                "fiction in English",
                "slavery",
                "romantic fiction",
                "Reading Level-Grade 11",
                "revenge in fiction",
                "Reading Level-Grade 12",
                "Psychological fiction"
            ]
```


The “value” can be string, number, array, Boolean, object…

```
{
    "subject_type": "subject",
    "name": "love",
    "key": "/subjects/love",
    "ebook_count": 10350,
    "works": [
        {
            "printdisabled": true,
            "cover_id": 8256292,
            "ia_collection": [
                "toronto",
                "JaiGyan",
                "digitallibraryindia",
                "americana",
                "duke_libraries",
                "delawarecountydistrictlibrary",
                "librivoxaudio",
                "gutenberg",
                "audio_bookspoetry",
                "china",
                "trent_university",
                "inlibrary",
                "printdisabled",
                "internetarchivebooks",
                "cornell",
                "librarygenesis",
                "opensource",
                "robarts",
                "europeanlibraries"
            ],
            "has_fulltext": true,
            "edition_count": 475,
            "checked_out": false,
            "title": "Wuthering Heights",
            "public_scan": true,
            "cover_edition_key": "OL26500373M",
            "lending_edition": "OL11072346M",
            "first_publish_year": null,
            "key": "/works/OL15035771W",
            "authors": [
                {
                    "name": "Emily Bront\u00eb",
                    "key": "/authors/OL4327048A"
                }
            ],
            "ia": "wuthering_heights_0801_librivox",
            "lending_identifier": "wutheringhgts00emil",
            "availability": {
                "status": "open",
                "openlibrary_work": null,
                "isbn": null,
                "num_waitlist": null,
                "last_loan_date": null,
                "openlibrary_edition": null,
                "oclc": null,
                "collection": "librivoxaudio,audio_bookspoetry,fav-
                "last_waitlist_date": null,
                "identifier": "wuthering_heights_0801_librivox"
            },
            "subject": [
                "Heathcliff (Fictitious character)",
                "Man-woman relationships",
                "love",
                "Accessible book",
                "Fiction",
                "England in fiction",
                "Literatuurkritiek",
                "Historical Fiction",
                "Orphans in fiction",
                "Foundlings",
                "Wuthering Heights (Bront\u00eb, Emily)",
                "death",
                "Reading Level-Grade 7",
                "Reading Level-Grade 9",
                "Reading Level-Grade 8",
                "Man-woman relationships in fiction",
                "romance",
                "French fiction",
                "Landscape in literature in fiction",
                "Country life in fiction",
                "Classic Literature",
                "foundlings in fiction",
                "Love in fiction",
                "Open Library Staff Picks",
                "Landscape in literature",
                "revenge",
                "Rejection (Psychology)",
                "Drama",
                "Country life",
                "Triangles (Interpersonal relations)",
                "Reading Level-Grade 10",
                "Manners and customs",
                "Rural families",
                "Rural families in fiction",
                "Social life and customs",
                "tragedy in fiction",
                "orphans",
                "fiction in English",
                "slavery",
                "romantic fiction",
                "Reading Level-Grade 11",
                "revenge in fiction",
                "Reading Level-Grade 12",
                "Psychological fiction"
            ]
        },
```


Object

```
{
    "subject_type": "subject",
    "name": "love",
    "key": "/subjects/love",
    "ebook_count": 10350,
    "works": [
        {
            "printdisabled": true,
            "cover_id": 8256292,
            "ia_collection": [
                "toronto",
                "JaiGyan",
                "digitallibraryindia",
                "americana",
                "duke_libraries",
                "delawarecountydistrictlibrary",
                "librivoxaudio",
                "gutenberg",
                "audio_bookspoetry",
                "china",
                "trent_university",
                "inlibrary",
                "printdisabled",
                "internetarchivebooks",
                "cornell",
                "librarygenesis",
                "opensource",
                "robarts",
                "europeanlibraries"
            ],
            "has_fulltext": true,
            "edition_count": 475,
            "checked_out": false,
            "title": "Wuthering Heights",
            "public_scan": true,
            "cover_edition_key": "OL26500373M",
            "lending_edition": "OL11072346M",
            "first_publish_year": null,
            "key": "/works/OL15035771W",
            "authors": [
                {
                    "name": "Emily Bront\u00eb",
                    "key": "/authors/OL4327048A"
                }
            ],
            "ia": "wuthering_heights_0801_librivox",
            "lending_identifier": "wutheringhgts00emil",
            "availability": {
                "status": "open",
                "openlibrary_work": null,
                "isbn": null,
                "num_waitlist": null,
                "last_loan_date": null,
                "openlibrary_edition": null,
                "oclc": null,
                "collection": "librivoxaudio,audio_bookspoetry,fav-
                "last_waitlist_date": null,
                "identifier": "wuthering_heights_0801_librivox"
            },
            "subject": [
                "Heathcliff (Fictitious character)",
                "Man-woman relationships",
                "love",
                "Accessible book",
                "Fiction",
                "England in fiction",
                "Literatuurkritiek",
                "Historical Fiction",
                "Orphans in fiction",
                "Foundlings",
                "Wuthering Heights (Bront\u00eb, Emily)",
                "death",
                "Reading Level-Grade 7",
                "Reading Level-Grade 9",
                "Reading Level-Grade 8",
                "Man-woman relationships in fiction",
                "romance",
                "French fiction",
                "Landscape in literature in fiction",
                "Country life in fiction",
                "Classic Literature",
                "foundlings in fiction",
                "Love in fiction",
                "Open Library Staff Picks",
                "Landscape in literature",
                "revenge",
                "Rejection (Psychology)",
                "Drama",
                "Country life",
                "Triangles (Interpersonal relations)",
                "Reading Level-Grade 10",
                "Manners and customs",
                "Rural families",
                "Rural families in fiction",
                "Social life and customs",
                "tragedy in fiction",
                "orphans",
                "fiction in English",
                "slavery",
                "romantic fiction",
                "Reading Level-Grade 11",
                "revenge in fiction",
                "Reading Level-Grade 12",
                "Psychological fiction"
            ]
        },
```


Array

```
{
    "subject_type": "subject",
    "name": "love",
    "key": "/subjects/love",
    "ebook_count": 10350,
    "works": [
        {
            "printdisabled": true,
            "cover_id": 8256292,
            "ia_collection": [
                "toronto",
                "JaiGyan",
                "digitallibraryindia",
                "americana",
                "duke_libraries",
                "delawarecountydistrictlibrary",
                "librivoxaudio",
                "gutenberg",
                "audio_bookspoetry",
                "china",
                "trent_university",
                "inlibrary",
                "printdisabled",
                "internetarchivebooks",
                "cornell",
                "librarygenesis",
                "opensource",
                "robarts",
                "europeanlibraries"
            ],
            "has_fulltext": true,
            "edition_count": 475,
            "checked_out": false,
            "title": "Wuthering Heights",
            "public_scan": true,
            "cover_edition_key": "OL26500373M",
            "lending_edition": "OL11072346M",
            "first_publish_year": null,
            "key": "/works/OL15035771W",
            "authors": [
                {
                    "name": "Emily Bront\u00eb",
                    "key": "/authors/OL4327048A"
                }
            ],
            "ia": "wuthering_heights_0801_librivox",
            "lending_identifier": "wutheringhgts00emil",
            "availability": {
                "status": "open",
                "openlibrary_work": null,
                "isbn": null,
                "num_waitlist": null,
                "last_loan_date": null,
                "openlibrary_edition": null,
                "oclc": null,
                "collection": "librivoxaudio,audio_bookspoetry,fav-
                "last_waitlist_date": null,
                "identifier": "wuthering_heights_0801_librivox"
            },
            "subject": [
                "Heathcliff (Fictitious character)",
                "Man-woman relationships",
                "love",
                "Accessible book",
                "Fiction",
                "England in fiction",
                "Literatuurkritiek",
                "Historical Fiction",
                "Orphans in fiction",
                "Foundlings",
                "Wuthering Heights (Bront\u00eb, Emily)",
                "death",
                "Reading Level-Grade 7",
                "Reading Level-Grade 9",
                "Reading Level-Grade 8",
                "Man-woman relationships in fiction",
                "romance",
                "French fiction",
                "Landscape in literature in fiction",
                "Country life in fiction",
                "Classic Literature",
                "foundlings in fiction",
                "Love in fiction",
                "Open Library Staff Picks",
                "Landscape in literature",
                "revenge",
                "Rejection (Psychology)",
                "Drama",
                "Country life",
                "Triangles (Interpersonal relations)",
                "Reading Level-Grade 10",
                "Manners and customs",
                "Rural families",
                "Rural families in fiction",
                "Social life and customs",
                "tragedy in fiction",
                "orphans",
                "fiction in English",
                "slavery",
                "romantic fiction",
                "Reading Level-Grade 11",
                "revenge in fiction",
                "Reading Level-Grade 12",
                "Psychological fiction"
            ]
        },
```


Array of objects

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - apis.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 apis.py

apis

apis.py

```
5  url = "http://openlibrary.org/subjects/love.json"
6  response = requests.get(url)
7
8  print(response.status_code)
9
10 data = response.json()
11
12 loveBooks = pd.json_normalize(data["works"], sep="_")
13 print(loveBooks.info())
```

Json_normalize is used to convert the json string to a pandas dataframe
Sep is the separator for nesting

Run: apis

```
Data columns (total 34 columns):
 #   Column                          Non-Null Count  Dtype
---  ------                          --------------  -----
 0   printdisabled                   12 non-null     bool
 1   cover_id                        12 non-null     int64
 2   ia_collection                   12 non-null     object
 3   has_fulltext                    12 non-null     bool
 4   edition_count                   12 non-null     int64
 5   checked_out                     12 non-null     bool
 6   title                           12 non-null     object
 7   public_scan                     12 non-null     bool
```

1: Project 7: Structure 2: Favorites

6: TODO 4: Run 5: Debug Terminal Python Console 2 Event Log

Packages installed successfully: Installed packages: 'requests' (18 minutes ago) 11:1 CRLF UTF-8 4 spaces Python 3.8 (venv)

```
RangeIndex: 12 entries, 0 to 11
Data columns (total 27 columns):
printdisabled                       12 non-null bool
cover_id                            12 non-null int64
ia_collection                       12 non-null object
has_fulltext                        12 non-null bool
edition_count                       12 non-null int64
checked_out                         12 non-null bool
title                               12 non-null object
public_scan                         12 non-null bool
cover_edition_key                   12 non-null object
lendinglibrary                      12 non-null bool
lending_edition                     12 non-null object
first_publish_year                  0 non-null object
key                                 12 non-null object
authors                             12 non-null object
ia                                  12 non-null object
lending_identifier                  12 non-null object
subject                             12 non-null object
availability_status                 12 non-null object
availability_openlibrary_work       11 non-null object
availability_isbn                   0 non-null object
availability_num_waitlist           0 non-null object
availability_last_loan_date         0 non-null object
availability_openlibrary_edition    11 non-null object
availability_oclc                   0 non-null object
availability_collection             12 non-null object
availability_last_waitlist_date     0 non-null object
availability_identifier             12 non-null object
dtypes: bool(5), int64(2), object(20)
memory usage: 2.2+ KB
None
```

```
{
    "subject_type": "subject",
    "name": "love",
    "key": "/subjects/love",
    "ebook_count": 10350,
    "works": [
        {
            "printdisabled": true,
            "cover_id": 8256292,
            "ia_collection": [
                "toronto",
                "JaiGyan",
                "digitallibraryindia",
                "americana",
                "duke_libraries",
                "delawarecountydistrictlibrary",
                "librivoxaudio",
                "gutenberg",
                "audio_bookspoetry",
                "china",
                "trent_university",
                "inlibrary",
                "printdisabled",
                "internetarchivebooks",
                "cornell",
                "librarygenesis",
                "opensource",
                "robarts",
                "europeanlibraries"
            ],
            "has_fulltext": true,
            "edition_count": 475,
            "checked_out": false,
            "title": "Wuthering Heights",
            "public_scan": true,
            "cover_edition_key": "OL26500373M",
            "lending_edition": "OL11072346M",
            "first_publish_year": null,
            "key": "/works/OL15035771W",
            "authors": [
                {
                    "name": "Emily Bront\u00eb",
                    "key": "/authors/OL4327048A"
                }
            ],
            "ia": "wuthering_heights_0801_librivox",
            "lending_identifier": "wutheringhgts00emil",
            "availability": {
                "status": "open",
                "openlibrary_work": null,
                "isbn": null,
                "num_waitlist": null,
                "last_loan_date": null,
                "openlibrary_edition": null,
                "oclc": null,
                "collection": "librivoxaudio,audio_bookspoetry,fav-
                "last_waitlist_date": null,
                "identifier": "wuthering_heights_0801_librivox"
            },
            "subject": [
                "Heathcliff (Fictitious character)",
                "Man-woman relationships",
                "love",
                "Accessible book",
                "Fiction",
                "England in fiction",
                "Literatuurkritiek",
                "Historical Fiction",
                "Orphans in fiction",
                "Foundlings",
                "Wuthering Heights (Bront\u00eb, Emily)",
                "death",
                "Reading Level-Grade 7",
                "Reading Level-Grade 7",
                "Reading Level-Grade 9",
                "Reading Level-Grade 8",
                "Man-woman relationships in fiction",
                "romance",
                "French fiction",
                "Landscape in literature in fiction",
                "Country life in fiction",
                "Classic Literature",
                "foundlings in fiction",
                "Love in fiction",
                "Open Library Staff Picks",
                "Landscape in literature",
                "revenge",
                "Rejection (Psychology)",
                "Drama",
                "Country life",
                "Triangles (Interpersonal relations)",
                "Reading Level-Grade 10",
                "Manners and customs",
                "Rural families",
                "Rural families in fiction",
                "Social life and customs",
                "tragedy in fiction",
                "orphans",
                "fiction in English",
                "slavery",
                "romantic fiction",
                "Reading Level-Grade 11",
                "revenge in fiction",
                "Reading Level-Grade 12",
                "Psychological fiction"
            ]
        },
```

```
RangeIndex: 12 entries, 0 to 11
Data columns (total 27 columns):
printdisabled                       12 non-null bool
cover_id                            12 non-null int64
ia_collection                       12 non-null object
has_fulltext                        12 non-null bool
edition_count                       12 non-null int64
checked_out                         12 non-null bool
title                               12 non-null object
public_scan                         12 non-null bool
cover_edition_key                   12 non-null object
lendinglibrary                      12 non-null bool
lending_edition                     12 non-null object
first_publish_year                  0 non-null object
key                                 12 non-null object
authors                             12 non-null object
ia                                  12 non-null object
lending_identifier                  12 non-null object
subject                             12 non-null object
availability_status                 12 non-null object
availability_openlibrary_work       11 non-null object
availability_isbn                   0 non-null object
availability_num_waitlist           0 non-null object
availability_last_loan_date         0 non-null object
availability_openlibrary_edition    11 non-null object
availability_oclc                   0 non-null object
availability_collection             12 non-null object
availability_last_waitlist_date     0 non-null object
availability_identifier             12 non-null object
dtypes: bool(5), int64(2), object(20)
memory usage: 2.2+ KB
None
```

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - apis.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 apis.py

apis.py

```
url = "http://openlibrary.org/subjects/love.json"
response = requests.get(url)

data = response.json()

loveBooks = \
    pd.json_normalize(data["works"], sep="_", record_path="authors", meta=["title","edition_count","cover_id"])

print(loveBooks.head(3))
```

Record_path is the path to each record that you want to get to. E.g. list *authors*.
Meta is the fields that you want to select for the dataframe

Run: apis

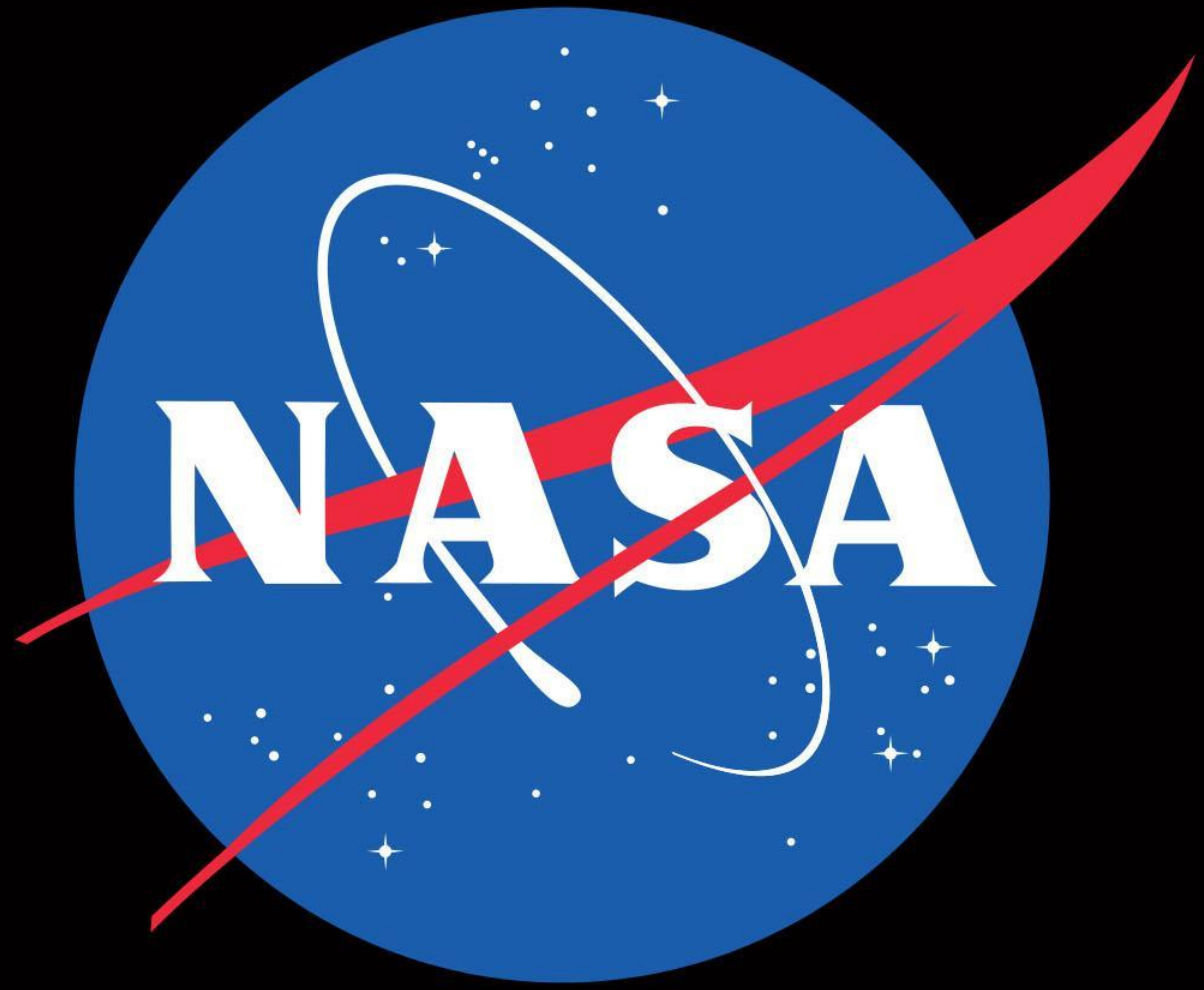
```
"C:\Users\Nitin Kale\venv\Scripts\python.exe" "C:/Users/Nitin Kale/PycharmProjects/ITP449 Master/Lecture 13/apis
              name                   key  ... edition_count cover_id
0    Emily Brontë  /authors/OL4327048A  ...           481  8254607
1           Plato   /authors/OL189658A  ...           252  8236248
2  Edmond Rostand    /authors/OL39281A  ...           198  8236320

[3 rows x 5 columns]

Process finished with exit code 0
```

6: TODO 4: Run 5: Debug Terminal Python Console Event Log

Packages installed successfully: Installed packages: 'requests' (22 minutes ago) 12:1 CRLF UTF-8 4 spaces Python 3.8 (venv)


NASA

API Keys

Some websites like to control who/what/how
data are being accessed via APIs

For this purpose API keys are issued to users (or even other programs).
The API key is used to limit what users can do, as well as track the usage.

Generate API Key

Sign up for an application programming interface (API) key to access and use web services available on the Data.gov developer network.

* Required fields

* First Name

* Last Name

Mail - Nitin Vasant Kale - Ou | Discussion Board – 20202_itp_44 | NASA Open APIs

api.nasa.gov

{ APIs }

Overview | Generate API Key | Authentication | Recover API Key | Browse APIs

Generate API Key

Your API key for **kale@usc.edu** is:

`9iIQUASWVW74CH1d6G9E1thWRPvIKes6Cf9r6qB9`

You can start using this key to make web service requests. Simply pass your key in the URL when making a web request. Here's an example:

https://api.nasa.gov/planetary/apod?**api_key=9iIQUASWVW74CH1d6G9E1thWRPvIKes6Cf9r6qB9**

For additional support, please contact us. When contacting us, please tell us what API you're accessing and provide the following account details so we can quickly find you:

Account Email: kale@usc.edu
Account ID: 99d55313-4a63-49be-b323-56d3477d8f85

Authentication

You do not need to authenticate in order to explore the NASA data. However, if you will be intensively using the APIs to, say, support a mobile application, then you should sign up for a NASA developer key.

Web Service Rate Limits

Limits are placed on the number of API requests you may make using your API key. Rate limits may vary by service, but the defaults are:

- Hourly Limit: 1,000 requests per hour

For each API key, these limits are applied across all api.nasa.gov API requests. Exceeding these limits will lead to your API key being temporarily blocked from making further requests. The block will automatically be lifted by waiting an hour. If you need higher rate limits,

APOD: Astronomy Picture of the Day

Asteroids NeoWs: Near Earth Object Web Service

DONKI: Space Weather Database Of Notifications, Knowledge, Information

Earth: Unlock the significant public investment in earth observation data

EONET: The Earth Observatory Natural Event Tracker

EPIC: Earth Polychromatic Imaging Camera

Exoplanet: Programmatic access to NASA's Exoplanet Archive database

GeneLab: Programmatic interface for GeneLab's public data repository website

Insight: Mars Weather Service API

Mars Rover Photos: Image data gathered by NASA's Curiosity, Opportunity, and Spirit rovers on Mars

NASA Image and Video Library: API to access the NASA Image and Video Library site at images.nasa.gov

TechTransfer: Patents, Software, and Tech Transfer Reports

Satellite Situation Center: System to cast geocentric spacecraft location information into a framework of (empirical) geophysical regions

SSD/CNEOS: Solar System Dynamics and Center for Near-Earth Object Studies

Techport: API to make NASA technology project data available in a machine-readable format

TLE API: Two line element data for earth-orbiting objects at a given point in time

EPIC DAILY "BLUE MARBLE" API

The EPIC API provides information on the daily imagery collected by DSCOVR's Earth Polychromatic Imaging Camera (EPIC) instrument. Uniquely positioned at the Earth-Sun Lagrange point, EPIC provides full disc imagery of the Earth and captures unique perspectives of certain astronomical events such as lunar transits using a 2048x2048 pixel CCD (Charge Coupled Device) detector coupled to a 30-cm aperture Cassegrain telescope.

Development of the EPIC API began in 2015, and is supported by the web development team for the Laboratory for Atmospheres in the Earth Sciences Division of the Goddard Space Flight Center.

API VERSION 2.0

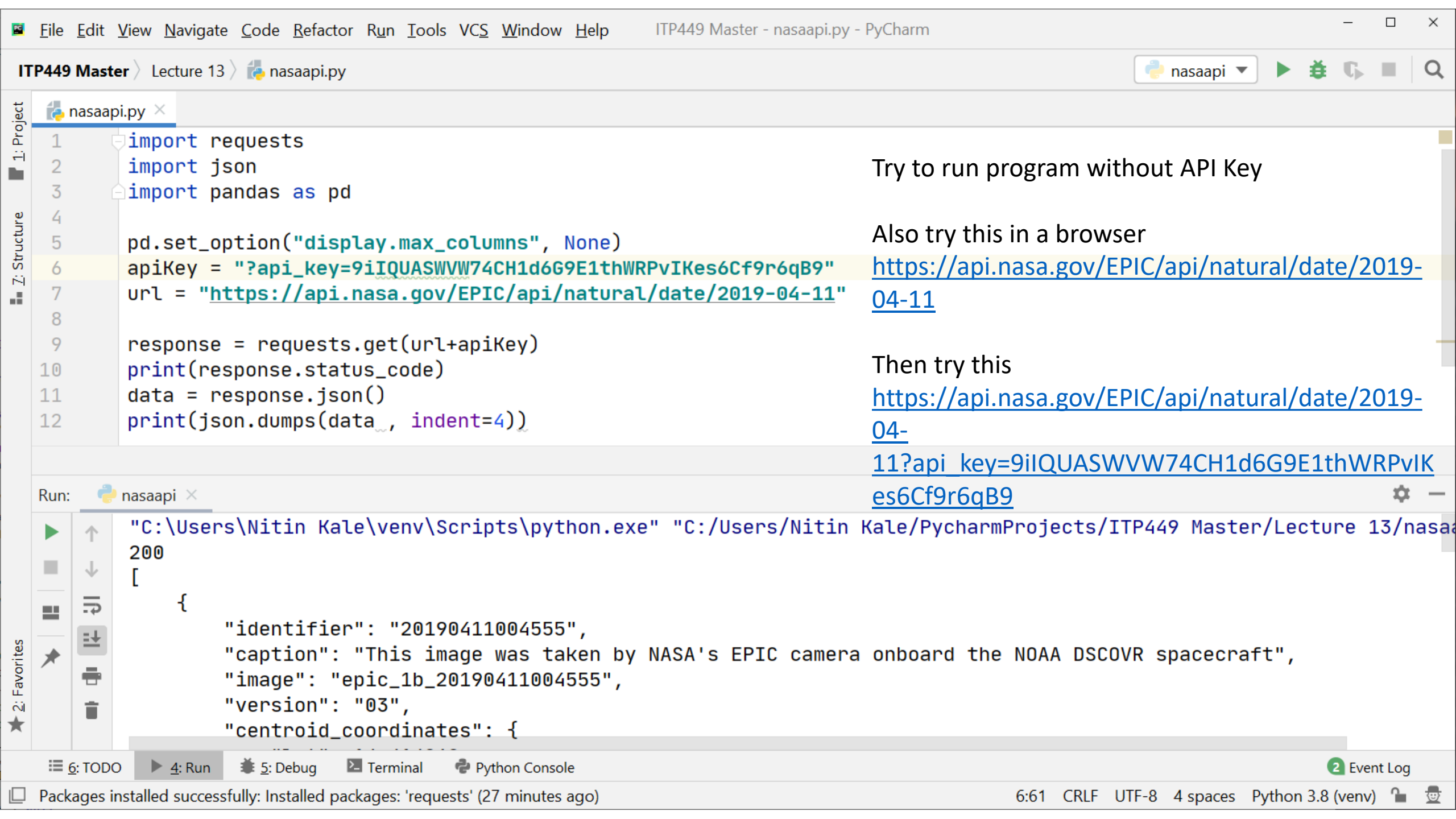
Image metadata and key information regarding the natural color and the enhanced color imagery are provided by the JSON API and can be requested by date and for the most recent available date. A listing of all available dates can also be retrieved via the API for more granular control.

QUERYING THE API

Parameter	Type	Default	Description
natural	string	Most Recent Natural Color	Retrieve metadata on the most recent date of natural color imagery.
natural/date	YYYY-MM-DD	Most Recent Available	Retrieve metadata for natural color imagery available for a given date.
natural/all	string	Dates for Natural Color	Retrieve a listing of all dates with available natural color imagery.
natural/available	string	Dates for Natural Color	Retrieve a listing of all dates with available natural color imagery.
enhanced	string	Most Recent Enhanced Color	Retrieve metadata on the most recent date of enhanced color imagery.
enhanced/date	YYYY-MM-DD	Most Recent Available	Retrieve metadata for enhanced color imagery for a given date.
enhanced/all	string	Dates for Enhanced Imagery	Retrieve a listing of all dates with available enhanced color imagery.
enhanced/available	string	Dates for Enhanced Imagery	Retrieve a listing of all dates with available enhanced color imagery.

For this new version of the API, we have tried to maintain as much backward compatibility as possible to the original version of the API. In addition to the parameters provided above, the majority of the original API parameters should work as well. That backward compatibility may be phased out over time, so please plan accordingly.

RETRIEVABLE METADATA


File Edit View Navigate Code Refactor Run Tools VCS Window Help
ITP449 Master - nasaapi.py - PyCharm
ITP449 Master 〉 Lecture 13 〉 nasaapi.py
nasaapi
nasaapi.py
1 import requests
2 import json
3 import pandas as pd
4
5 pd.set_option("display.max_columns", None)
6 apiKey = "?api_key=9iIQUASWVW74CH1d6G9E1thWRPvIKes6Cf9r6qB9"
7 url = "https://api.nasa.gov/EPIC/api/natural/date/2019-04-11"
8
9 response = requests.get(url+apiKey)
10 print(response.status_code)
11 data = response.json()
12 print(json.dumps(data , indent=4))
Try to run program without API Key
Also try this in a browser
https://api.nasa.gov/EPIC/api/natural/date/2019-04-11
Then try this
https://api.nasa.gov/EPIC/api/natural/date/2019-04-11?api_key=9iIQUASWVW74CH1d6G9E1thWRPvIKes6Cf9r6qB9
Run: nasaapi
"C:\Users\Nitin Kale\venv\Scripts\python.exe" "C:/Users/Nitin Kale/PycharmProjects/ITP449 Master/Lecture 13/nasa
200
[
{
"identifier": "20190411004555",
"caption": "This image was taken by NASA's EPIC camera onboard the NOAA DSCOVR spacecraft",
"image": "epic_1b_20190411004555",
"version": "03",
"centroid_coordinates": {
6: TODO
4: Run
5: Debug
Terminal
Python Console
Event Log
Packages installed successfully: Installed packages: 'requests' (27 minutes ago)
6:61 CRLF UTF-8 4 spaces Python 3.8 (venv)

Demo

Plot the global temperature anomaly from 1880 to today.

You can get the data here using an API
https://public.opendatasoft.com/explore/dataset/global-temperature-time-series/api/?rows=200

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - temperature.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 temperature.py

temperature

temperature.py

```
1  import requests
2  import json
3  import pandas as pd
4  import matplotlib.pyplot as plt
5
6  pd.set_option("display.max_columns", None)
7
8  response = \
9      requests.get\
10     ("https://public.opendatasoft.com/api/records/1.0/search/?dataset=global-temperature-time-series&q=&rows=200")
11
12 data = response.json()
```

Run: temperature

```
"C:\Users\Nitin Kale\venv\Scripts\python.exe" "C:/Users/Nitin Kale/PycharmProjects/ITP449 Master/Lecture 13/temperatur
                          datasetid                                  recordid  \
0    global-temperature-time-series  249d72f6ad4d6e70497954cb1de28547d9d26781
1    global-temperature-time-series  88780a512e268f78594d21913894e192bcfc9c8c
2    global-temperature-time-series  7f1ef79a8f64d2205ea048e8ef2e63fed7b0eebf
3    global-temperature-time-series  49c7c5d49269ced54d1d6cb69e541a7c4aa0702a
4    global-temperature-time-series  50ab67f9da6104cb136b3c6edd13f700a1eac26a
..                              ...                                      ...
195  global-temperature-time-series  d23d3106d654abafc1c545cb2ad2bb027f9546c3
```

6: TODO 4: Run 5: Debug Terminal Python Console Event Log

Packages installed successfully: Installed packages: 'requests' (36 minutes ago) 12:23 CRLF UTF-8 4 spaces Python 3.8 (venv)

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - temperature.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 temperature.py

temperature

temperature.py

```
13
14  temp = pd.json_normalize(data["records"], sep="_")
15
16  temp = temp.drop(columns=["record_timestamp","fields_source"], axis=1)
17  temp["fields_date"] = pd.to_datetime(temp["fields_date"])
18
19  temp.sort_values(by=["fields_date"], inplace=True)
20  print(temp)
```

Run: temperature

```
179  1880-01-01      -0.3000
0    1880-12-01      -0.2200
107  1882-12-01      -0.1705
178  1884-01-01      -0.1843
177  1884-08-01      -0.2700
..          ...         ...
160  2013-07-01       0.6662
29   2015-04-01       0.7400
193  2016-02-01       1.1921
132  2016-08-01       0.8998
```

6: TODO 4: Run 5: Debug Terminal Python Console 2 Event Log

PEP 8: W292 no newline at end of file 20:12 CRLF UTF-8 4 spaces Python 3.8 (venv)

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449 Master - temperature.py - PyCharm

ITP449 Master 〉 Lecture 13 〉 temperature.py

temperature

temperature.py

```
13
14  temp = pd.json_normalize(data["records"], sep="_")
15
16  temp = temp.drop(columns=["record_timestamp","fields_source"], axis=1)
17  temp["fields_date"] = pd.to_datetime(temp["fields_date"])
18
19  temp.sort_values(by=["fields_date"], inplace=True)
20  print(temp)
21
22  plt.plot(temp.fields_date, temp.fields_mean, marker = "o")
23  plt.show()
```

Run: temperature

```
193  2016-02-01       1.1921
132  2016-08-01       0.8998
173  2016-11-01       0.7504
226  2016-12-01       0.8100

[500 rows x 4 columns]

Process finished with exit code 0
```

1: Project 7: Structure 2: Favorites

6: TODO 4: Run 5: Debug Terminal Python Console 2 Event Log

Packages installed successfully: Installed packages: 'requests' (41 minutes ago) 19:51 CRLF UTF-8 4 spaces Python 3.8 (venv)

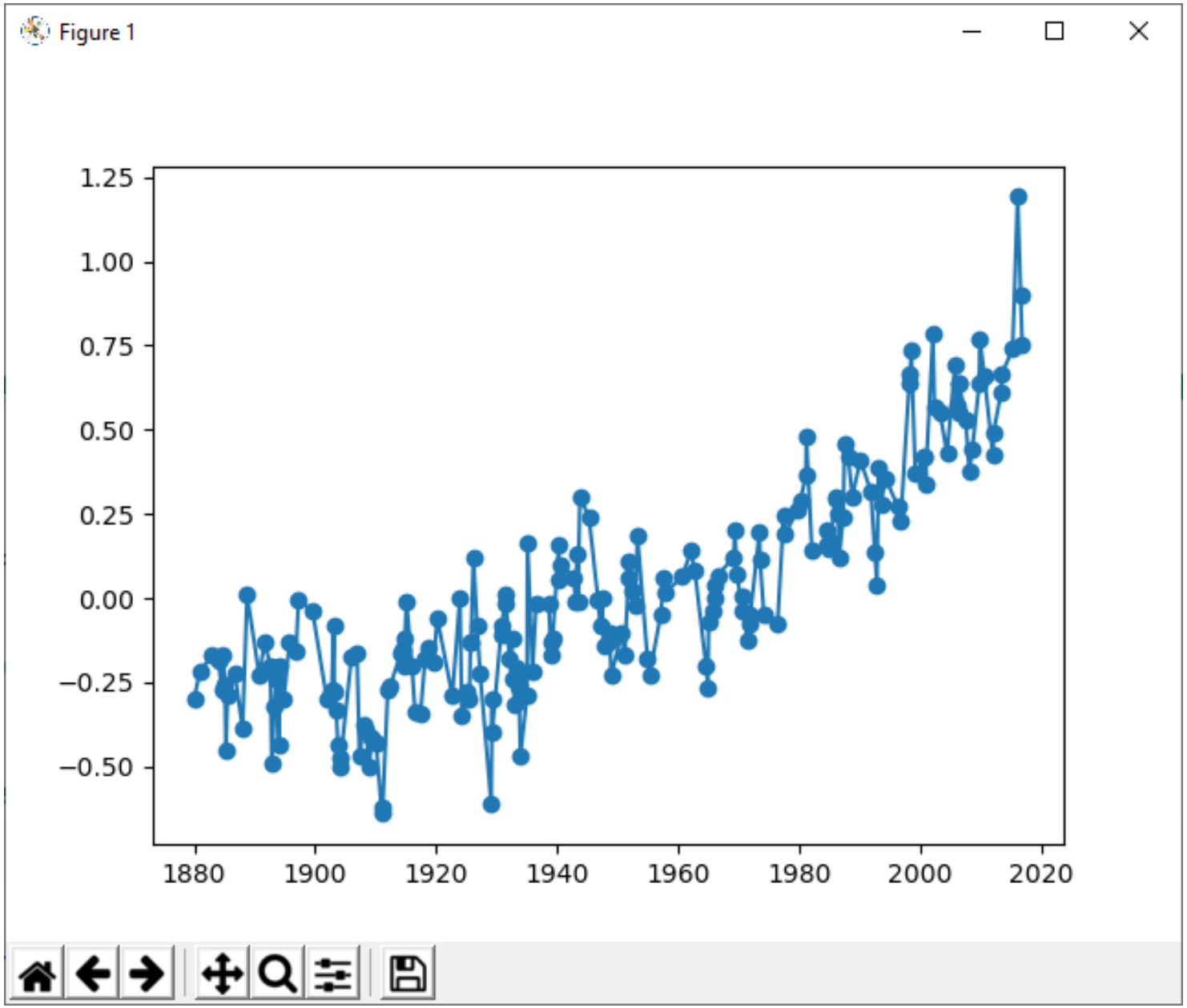


Figure 1
1.25
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
1880
1900
1920
1940
1960
1980
2000
2020

Forecasting

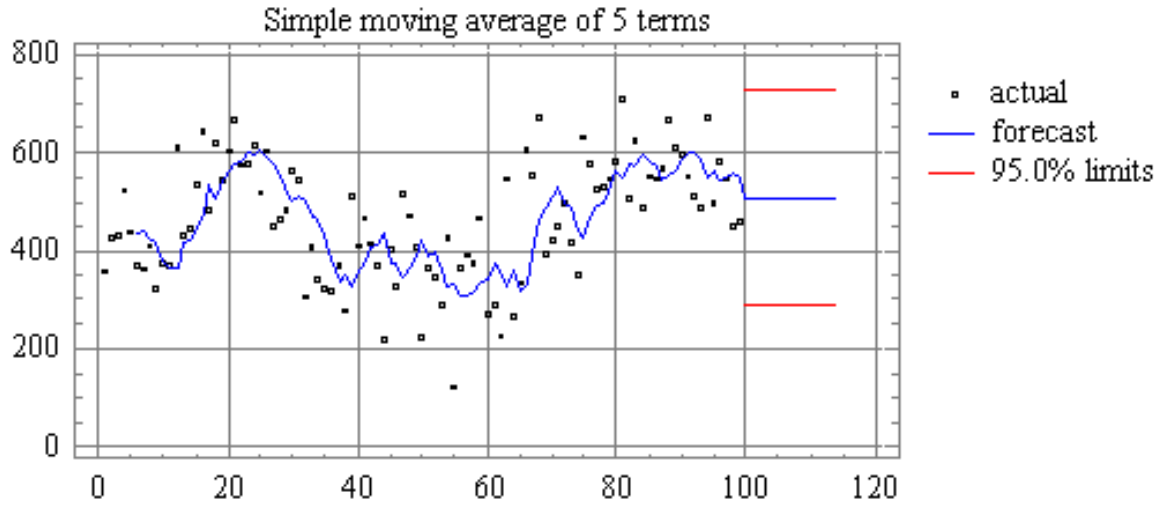
Time series forecasting

- A simple way to forecast the value of variable into the future is to use the mean.
- It assumes that the future is the average of the history
- Another approach is to use the average of the recent past – moving average
- It is called “smoothing” because the local averaging smooths out the fluctuations in the time series.
- The degree of smoothing can be controlled by the using a larger time frame of smoothing (number of terms in the smoothing)

Simple Moving Average

- Future value of Y (forecast for time t+1) is the simple average of the past m values.
- Note that after a certain number of future forecasts, the value of y is constant.
- These models do not consider trend.

$$\hat{Y}_{t+1} = \frac{Y_t + Y_{t-1} + \ldots + Y_{t-m+1}}{m}$$

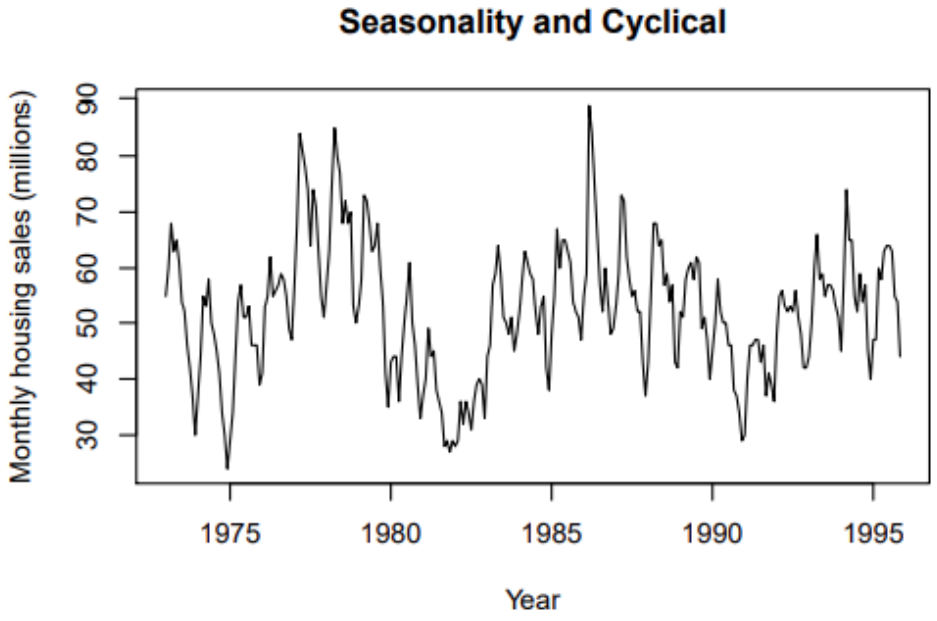


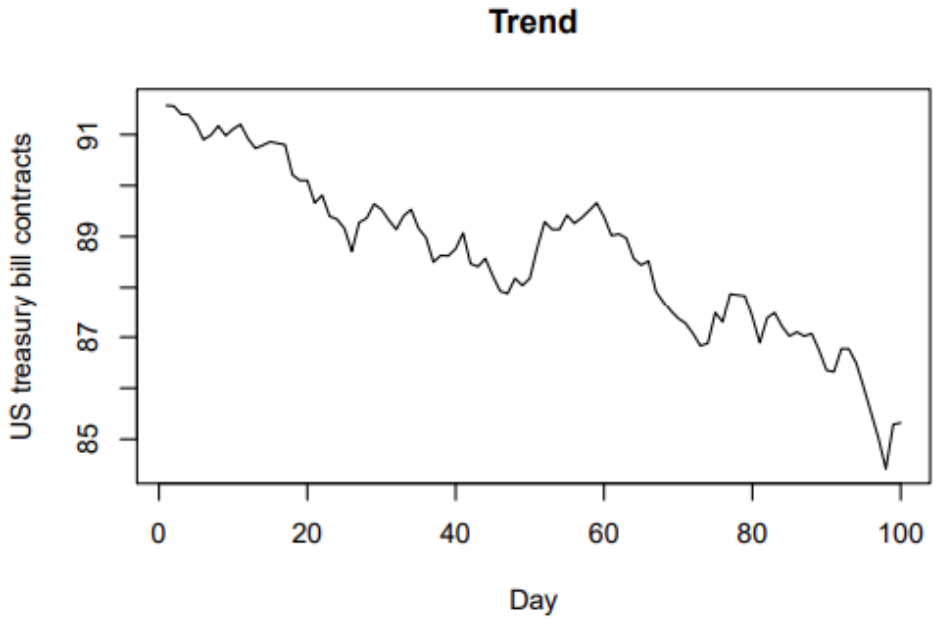
Trend, seasonality, random

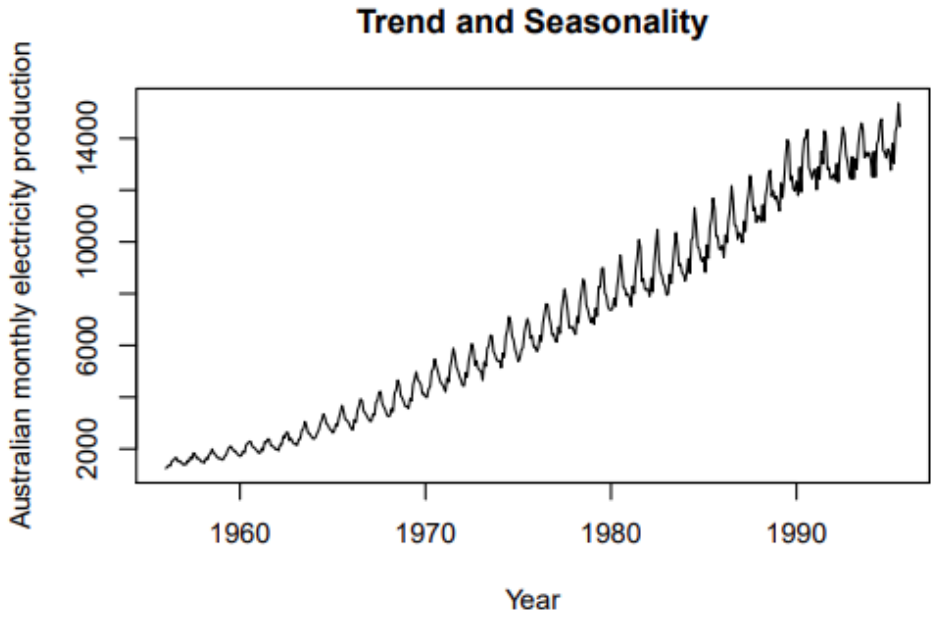
- A time series can be decomposed into four components
- Trend, seasonality, irregular (random), cyclical
- Trend – long-term increase or decrease
- Seasonality – regular fluctuations based on season (day/month/year etc.)
- Cyclical: non-fixed period rise and fall.
- Irregular/random: The leftover component after accounting for trend, seasonality, cyclical.

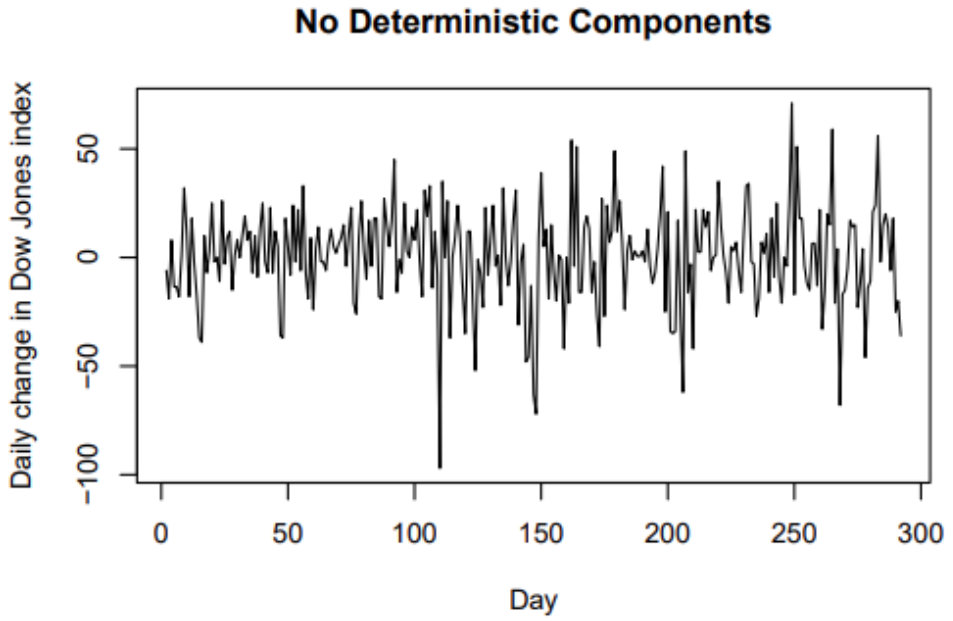
Stationarity

- Stationarity is the property of a time series where the mean, variance, autocorrelation are constant over time.
- Once a non-stationary time series has been made stationary, its forecast can be made using a technique such as moving averages


Seasonality and Cyclical
Monthly housing sales (millions)
30
40
50
60
70
80
90
1975
1980
1985
1990
1995
Year


Trend
US treasury bill contracts
85
87
89
91
0
20
40
60
80
100
Day


Trend and Seasonality
Australian monthly electricity production
2000
6000
10000
14000
1960
1970
1980
1990
Year


No Deterministic Components
Daily change in Dow Jones index
−100
−50
0
50
0
50
100
150
200
250
300
Day

Exponential Moving Average

- In simple moving average, we treat latest observations all equally.
- There is not discount on older observations. In other words, old observations have the same influence on the forecast as recent observations.
- In Exponential smoothing methods, we add a weighing parameter α
- When α = 0, we get simple moving average.

$$\hat{Y}_{t+1} = \alpha Y_t + (1-\alpha)\hat{Y}_t$$

$$\hat{Y}_{t+1} = \alpha[Y_t + (1-\alpha)Y_{t-1} + (1-\alpha)^2 Y_{t-2} + (1-\alpha)^3 Y_{t-3} + \ldots]$$

Double Exponential Smoothing

- If there is trend in the data, then we use double exponential smoothing
- Two smoothing parameters are used. α and β (the trend-smoothing parameter)

$$\hat{Y}_{t+k} = L_t + kT_t$$

$$L_t = \alpha Y_t + (1-\alpha)(L_{t-1} + T_{t-1})$$

$$T_t = \beta(L_t - L_{t-1}) + (1-\beta)T_{t-1}$$

Triple Exponential Smoothing

- For time series that has both trend and seasonality

A third weighting factor γ is introduced. This it the smoothing factor for *seasonality*.

Alpha, beta and gamma are iterated. At each iteration you compute the error from the observed values (as a sum of squares). The combination of alpha, beta and gamma the yield the lowest error is selected as the best fit.

If gamma = 0, then it is equivalent to double exponential smoothing.

If gamma =0 and beta =0 then it is equivalent to single exponential smoothing.

Triple Exponential Smoothing

$$l_x = \alpha(y_x - s_{x-L}) + (1-\alpha)(l_{x-1} + b_{x-1})$$
$$b_x = \beta(l_x - l_{x-1}) + (1-\beta)b_{x-1}$$
$$s_x = \gamma(y_x - l_x) + (1-\gamma)bs_{x-L}$$
$$\hat{y}_{x+m} = l_x + mb_x + s_{x-L+1+(m-1)modL}$$

Forecast temperature for the next
2 years using triple exponential smoothing

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449Fall2020noon - temp.py - PyCharm

ITP449Fall2020noon 〉 Lecture 13 〉 temp.py

temp

1: Project

7: Structure

temp.py

```
1  import requests
2  import json
3  import pandas as pd
4  import matplotlib.pyplot as plt
5
6  pd.set_option("max_columns", None)
7  url = \
8      "https://public.opendatasoft.com/api/records/1.0/search/?dataset=global-temperature-time-series&q=&rows=3288&" \
9      "refine.source=GISTEMP"
10
11 response = requests.get(url)
12
13 data = response.json()
14
15 tempDf = pd.json_normalize(data["records"], sep="_")
16 tempDf = tempDf.drop(columns=["recordid", "fields_source"], axis=1)
17
18 tempDf["fields_date"] = pd.to_datetime(tempDf["fields_date"])
19 tempDf.sort_values(by=["fields_date"], inplace=True)
20
```

Run: temp

```
  warnings.warn('No frequency information was'
C:\Users\Nitin Kale\venv\lib\site-packages\statsmodels\tsa\holtwinters\model.py:427: FutureWarning: After 0.13 initia
  warnings.warn(

Process finished with exit code 0
```

2: Favorites

6: TODO 4: Run Terminal Python Console Event Log

12:1 CRLF UTF-8 4 spaces Python 3.8 (venv)

File Edit View Navigate Code Refactor Run Tools VCS Window Help ITP449Fall2020noon - temp.py - PyCharm

ITP449Fall2020noon 〉 Lecture 13 〉 temp.py

temp

temp.py

```
20
21  tempDf.set_index("fields_date", inplace=True)
22
23  # plt.plot(tempDf.fields_date, tempDf.fields_mean, marker="o")
24
25  from statsmodels.tsa.holtwinters import ExponentialSmoothing
26  model = ExponentialSmoothing(tempDf.fields_mean, trend="add", damped_trend=True, seasonal="add", seasonal_periods=12)
27  tempFit = model.fit(optimized=True)
28
29  tempPred = tempFit.forecast(24)
30  plt.plot(tempDf.fields_mean, label="Actual Temperature")
31  plt.plot(tempFit.fittedvalues, '--', label = "Fitted temperature")
32  plt.plot(tempPred, "g", label="Forecasted temp")
33  plt.legend()
34  plt.show()
35
```

Run: temp

```
  warnings.warn('No frequency information was'
C:\Users\Nitin Kale\venv\lib\site-packages\statsmodels\tsa\holtwinters\model.py:427: FutureWarning: After 0.13 initia
  warnings.warn(

Process finished with exit code 0
```

6: TODO 4: Run Terminal Python Console Event Log

31:67 CRLF UTF-8 4 spaces Python 3.8 (venv)

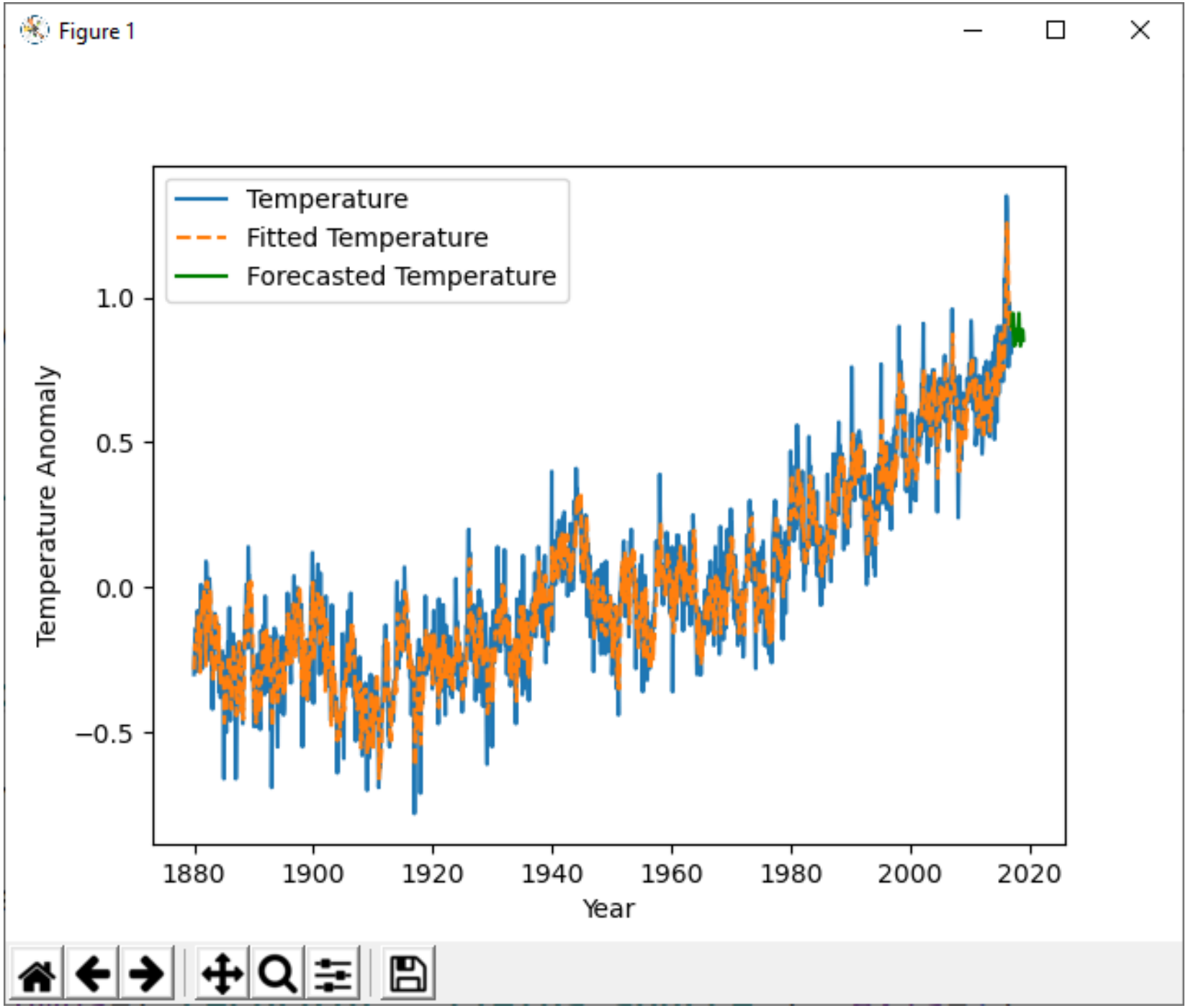


Figure 1
Temperature
Fitted Temperature
Forecasted Temperature
Temperature Anomaly
1.0
0.5
0.0
−0.5
1880
1900
1920
1940
1960
1980
2000
2020
Year