PyCharm
Virtual Environment
Dependency Management

PyCharm

PyCharm

1. Jupyter Notebook kullanılabiliyor.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması
10. Chunk mantığı ile çalışabiliyor olması.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması
10. Chunk mantığı ile çalışabiliyor olması.
11. Debug özelliği. Belirli noktalara debug pointler ekleyip o noktalardaki kod davranışları incelenebiliyor.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması
10. Chunk mantığı ile çalışabiliyor olması.
11. Debug özelliği. Belirli noktalara debug pointler ekleyip o noktalardaki kod davranışları incelenebiliyor.
12. Dosya dizin işlemlerinin çok daha kolay olması.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması
10. Chunk mantığı ile çalışabiliyor olması.
11. Debug özelliği. Belirli noktalara debug pointler ekleyip o noktalardaki kod davranışları incelenebiliyor.
12. Dosya dizin işlemlerinin çok daha kolay olması.
13. Path yakalama, istenilen dizinde terminal ya da klasör açabilme.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması
10. Chunk mantığı ile çalışabiliyor olması.
11. Debug özelliği. Belirli noktalara debug pointler ekleyip o noktalardaki kod davranışları incelenebiliyor.
12. Dosya dizin işlemlerinin çok daha kolay olması.
13. Path yakalama, istenilen dizinde terminal ya da klasör açabilme.
14. Otomatik olarak PEP8 kontrolü yapıyor olması.

PyCharm

1. Jupyter Notebook kullanılabiliyor.
2. Veri ya da grafik gibi çıktıları da verebiliyor.
3. Oluşan dataframe'leri print etmeden dataview'da görebiliyoruz. Büyük küçük değerler highlight edilebiliyor.
4. Terminal elimizin altında.
5. Git entegrasyonu ve versiyon kontrol-takibi. Tek tıkla repo işlemleri ve değişikliklerin arayüzde görülmesi imkanı.
6. Readme.md, make, requirements.txt gibi birçok özel dosya formatlarını desteklemesi etkin kullanım için yönlendirmesi.
7. Virtual environment ve dependency managment'in kolay olması.
8. Re-factor işleminin kolay olması.
9. Obje takibini çok kolay yapabiliyor olması
10. Chunk mantığı ile çalışabiliyor olması.
11. Debug özelliği. Belirli noktalara debug pointler ekleyip o noktalardaki kod davranışları incelenebiliyor.
12. Dosya dizin işlemlerinin çok daha kolay olması.
13. Path yakalama, istenilen dizinde terminal ya da klasör açabilme.
14. Otomatik olarak PEP8 kontrolü yapıyor olması.
15. Inspect code bölümüyle tüm kodları tarayıp hata raporlaması yapabiliyor olması. (Code -> Inspect Code)

PyCharm ile Yeni Proje Oluşturmak

1. Sıfırdan ortam oluşturma
2. Var olan ortamı seçme

Mac için: Interpreter -> Conda -> /Users/mvahit/anaconda3/bin/python3

Windows için: Anaconda prompt'u açtıktan sonra where python

Virtual Environments

Virtual Environment Nedir?

Farklı projeler için farklı ortamlar oluşturabilmenin yollarından birisidir.

Virtual Environment Nedir?

İyi de neden böyle bir şey

isteyelim?

Virtual Environment Nedir?

Birbirinden farklı ihtiyaçlara sahip olabilecek projeleri izole etmek için.

Örnek: pygame ile Python’da oyun yazmak istiyorsunuz diyelim.

Örnek: pygame ile Python’da oyun yazmak istiyorsunuz diyelim.

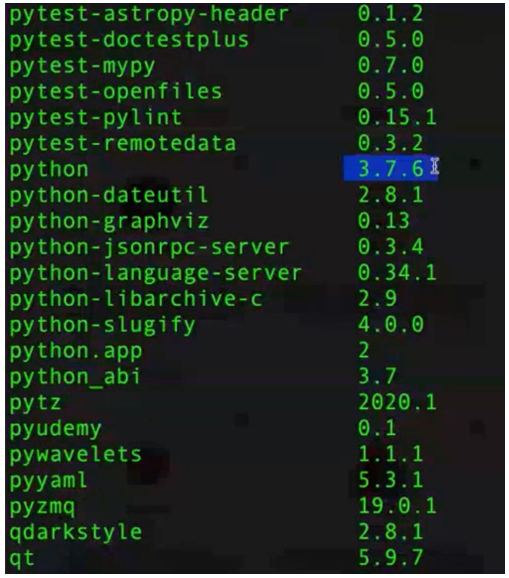
pygame der ki: ben 3 serisi ile çalışmayı bilmiyorum 2 serisi ile çalışabiliyorum (artık biliyor önceden bilmiyordu)

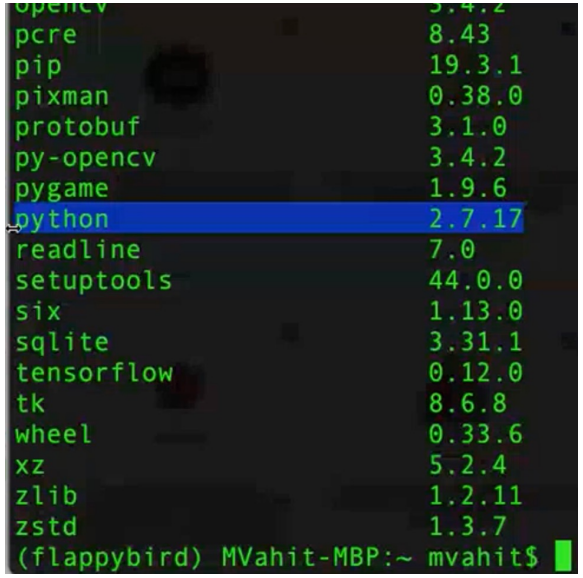
Örnek: pygame ile Python’da oyun yazmak istiyorsunuz diyelim.

pygame der ki: ben 3 serisi ile çalışmayı bilmiyorum 2 serisi ile çalışabiliyorum (artık biliyor önceden bilmiyordu)
Biz de diyoruz ki: iyi de karşim bizim bütün uygulamalarımız 3 serisi ile çalışıyor. Senin için 2’ye dönecek halimiz yok.

Örnek: pygame ile Python’da oyun yazmak istiyorsunuz diyelim.

pygame der ki: ben 3 serisi ile çalışmayı bilmiyorum 2 serisi ile çalışabiliyorum (artık biliyor önceden bilmiyordu)
Biz de diyoruz ki: iyi de karşim bizim bütün uygulamalarımız 3 serisi ile çalışıyor. Senin için 2’ye dönecek halimiz yok.
conda der ki: mrb

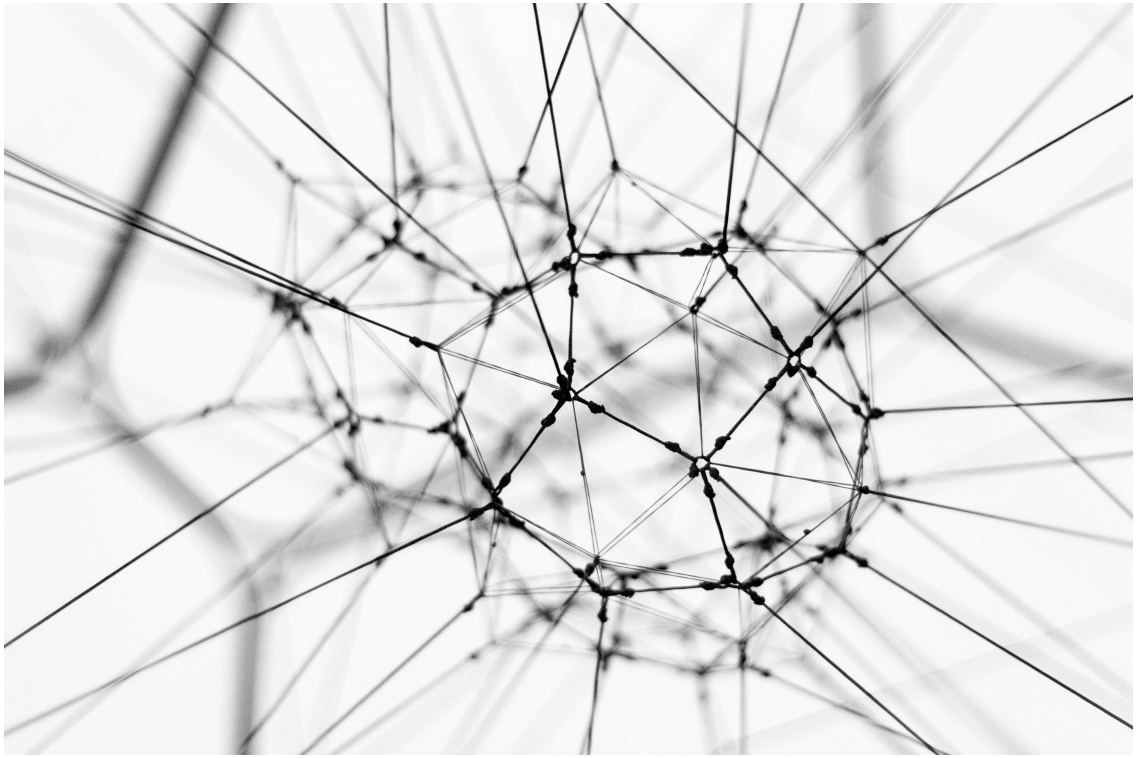




Virtual environments oluşturma ve yönetme işlevi olan araçlar:

- venv (Part of the standard library)
- virtualenv (Widely-used)
- pipenv (High-level interface)
- **conda** (Not only for Python)

Dependency Management



Dependency Management Nedir?

- Bağlılıkların yönetimi
- Paket yönetimi

İyi de ben hiç böyle bir şey ile karşılaşmadım?

İyi de ben hiç böyle bir şey ile karşılaşmadım?

Demek ki iyi yönetiyorlar ☺

- pip (requirements.txt)
- pipenv (Pipfile)
- conda (environment.yml)

Virtual Environment’ler ile Paket Yönetim Araçlarının ilişkisi Nedir?

Virtual Environment’ler ile Paket Yönetim Araçlarının ilişkisi Nedir?

- venv (Part of the standard library)
- virtualenv (Widely-used)
- pipenv (High-level interface)
- **conda** (Not only for Python)

Virtual Environment’ler ile Paket Yönetim Araçlarının ilişkisi Nedir?

- venv (Part of the standard library)
- virtualenv (Widely-used)
- pipenv (High-level interface)
- **conda** (Not only for Python)

- venv ve virtualenv paket yönetim aracı olarak pip'i kullanıyor.

Virtual Environment’ler ile Paket Yönetim Araçlarının ilişkisi Nedir?

- venv (Part of the standard library)
- virtualenv (Widely-used)
- pipenv (High-level interface)
- **conda** (Not only for Python)

- venv ve virtualenv paket yönetim aracı olarak pip'i kullanıyor.
- conda ve pipenv hem paket yönetimi hem virtual environment yönetimi yapabiliyor.

Sonuç?

- Conda paket yönetimi ve virtual environment yönetimi için kullanılabilir.
- pip paket yönetimi için kullanılabilir.
- Genelde bu ikisi kullanılır.

Yapacaklarımızla direk ilişkili mi?

Şimdilik değil. Son ayımız ve son haftalarımıza kadar ciddi bir işimiz olmayacak.

pip: pypi (python package index) paket yönetim aracı

Paket Yükleme:

- from PyPI: pip install requests
- from a local wheel file: pip install requests-2.22.0-py2.py3-none-any.whl
- from a Git repository: pip install git+https://github.com/psf/requests.git
- from a directory: pip install /home/user/src/requests

Versiyonlara Göre Paket Yükleme:

- Install specific version: pip install requests==2.22.0
- Install most recent version in a range: pip install requests>=2.22.0,<3
- Install package, avoid a specific version: pip install requests!=2.21.0

Mevcut Ortamdaki Paketlerin Versiyonlarını Export Etme:

- pip freeze > requirements.txt

Ortama Kayıtlı Paket Versiyonlarını Import Etme:

- pip install -r requirements.txt

conda: paket ve venv yönetim aracı (anaconda repository)

Install package:

conda install numpy

Install packages:

conda install numpy scipy pandas

Install package with spesific version:

conda install numpy=1.10

Remove package:

conda remove package_name

Listing installed packages:

conda list

Upgrade package:

conda upgrade conda

Upgrade all packages:

conda upgrade –all

Update current package:

conda update package_name

Update all current package:

conda update –all

Search package:

conda search *search_term*

Entering an environment:

conda activate my_env

Deactivate an environment:

conda deactivate

Listing environments

conda env list

Creating environment with spesific version of python:

conda create -n py3 python=3

Creating environment with spesific Python version and packages:

conda create -n mvk python=3 pandas numpy

Creating environment with packages:

conda create -n env_name list of packages

To list all of the packages in a deactivated environment:

conda list -n myenv

Removing environments

conda env remove -n env_name

Saving the packages to a YAML file

conda env export > environment.yaml

Loading yaml file

conda env create -f environment.yaml

Uygulama

- conda env list
- conda create -n my-env
- conda activate my-env
- conda list
- conda install numpy
- conda list
- pip install pandas
- conda list
- conda deactivate
- conda env remove -n myenv
- conda env list