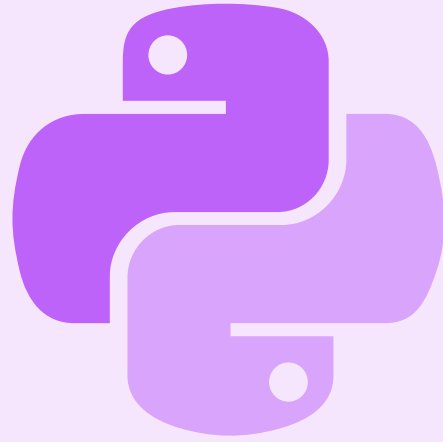


NESNE TABANLI PROGRAMLAMA - BIL207



Sunum Ana Hatları

BUGÜNÜN BAŞLIKLARI

- Python Geliştirme Ortamları
- Python Sınıflar

Python IDE

Bir IDE (veya Entegre Geliştirme Ortamı), yazılım geliştirmeye adanmış bir programdır. Adından da anlaşılacağı gibi, IDE'ler özellikle yazılım geliştirme için tasarlanmış birkaç aracı entegre eder. Bu araçlar genellikle şunları içerir:

- Kodu işlemek için tasarlanmış bir düzenleyici (örneğin, sözdizimi vurgulama ve otomatik tamamlama ile)
- Oluşturma, yürütme ve hata ayıklama araçları
- Bazı kaynak denetimi biçimleri

Çoğu IDE, birçok farklı programlama dilini destekler ve daha birçok özellik içerir. Bu nedenle, büyük olabilirler ve indirip kurmaları zaman alabilir. Bunları doğru bir şekilde kullanmak için ileri düzeyde bilgiye de ihtiyacınız olabilir. Çoğu iyi kod düzenleyici, kodu yürütebilir ve bir hata ayıklayıcıyı kontrol edebilir.

Python Destekli Genel Editörler ve IDE'ler

Eclipse + PyDev

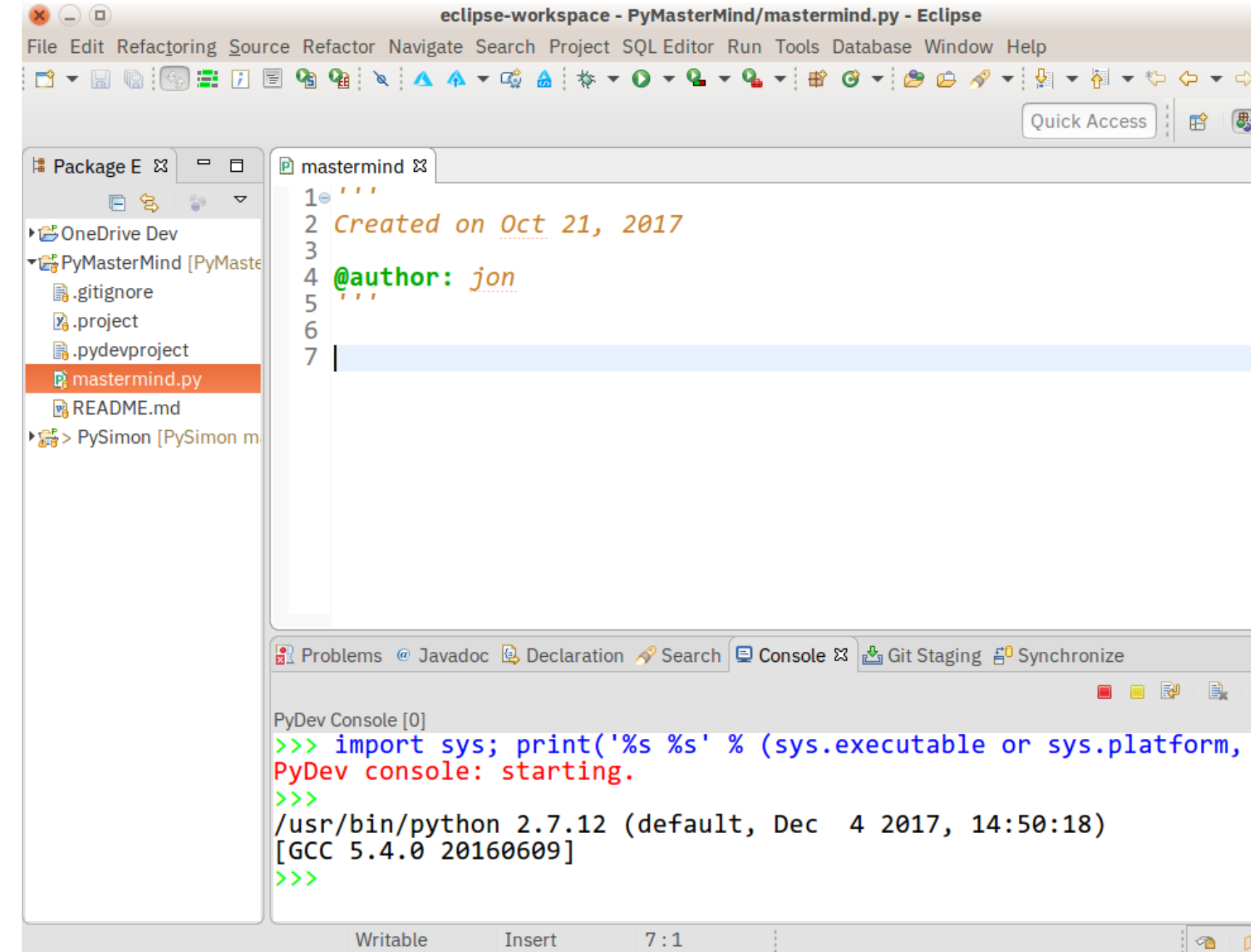
Kategori: IDE

Website: www.eclipse.org

Python araçları: PyDev, www.pydev.org

Artıları: Eclipse'i zaten yüklediyseniz, PyDev'i eklemek daha hızlı ve daha kolay olacaktır.

Eksileri: Python ile veya genel olarak yazılım geliştirme ile yeni başlıyorsanız, Eclipse ile başa çıkmak için çok şey olabilir.



Python Destekli Genel Editörler ve IDE'ler

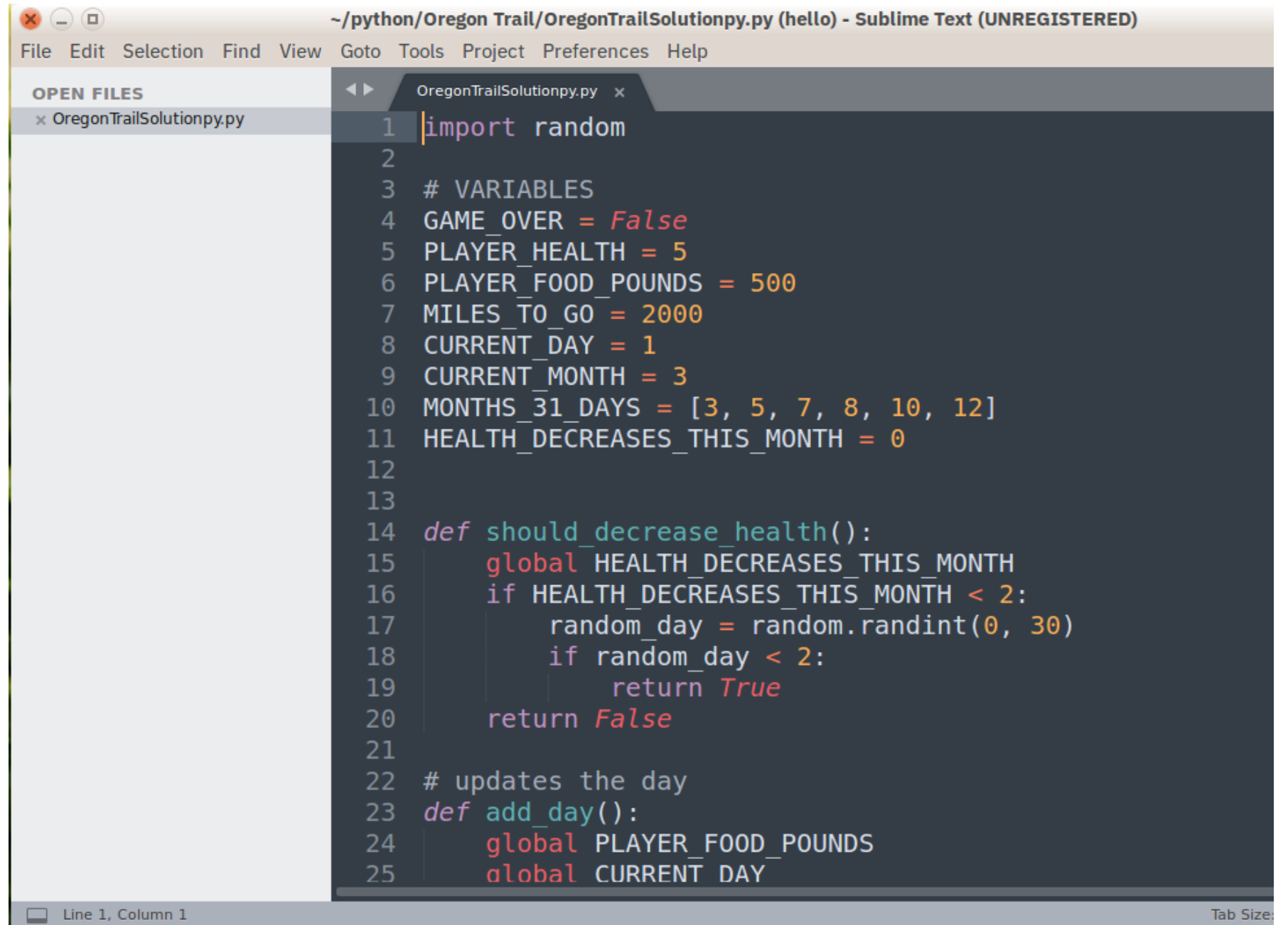
Sublime Text

Kategori: Code Editor

Website: <http://www.sublimetext.com>

Daha iyi bir metin editörü hayaliyle bir Google mühendisi tarafından yazılan Sublime Text, son derece popüler bir kod editörüdür.

Eksileri: Ücretsiz değildir, ancak değerlendirme sürümünü belirsiz bir süre için kullanabilirsiniz. Uzantıları yüklemek zor olabilir.



```
~/python/Oregon Trail/OregonTrailSolution.py (hello) - Sublime Text (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help

OPEN FILES
x OregonTrailSolution.py

1 import random
2
3 # VARIABLES
4 GAME_OVER = False
5 PLAYER_HEALTH = 5
6 PLAYER_FOOD_POUNDS = 500
7 MILES_TO_GO = 2000
8 CURRENT_DAY = 1
9 CURRENT_MONTH = 3
10 MONTHS_31_DAYS = [3, 5, 7, 8, 10, 12]
11 HEALTH_DECREASES_THIS_MONTH = 0
12
13
14 def should_decrease_health():
15     global HEALTH_DECREASES_THIS_MONTH
16     if HEALTH_DECREASES_THIS_MONTH < 2:
17         random_day = random.randint(0, 30)
18         if random_day < 2:
19             return True
20     return False
21
22 # updates the day
23 def add_day():
24     global PLAYER_FOOD_POUNDS
25     global CURRENT_DAY
```

Python Destekli Genel Editörler ve IDE'ler

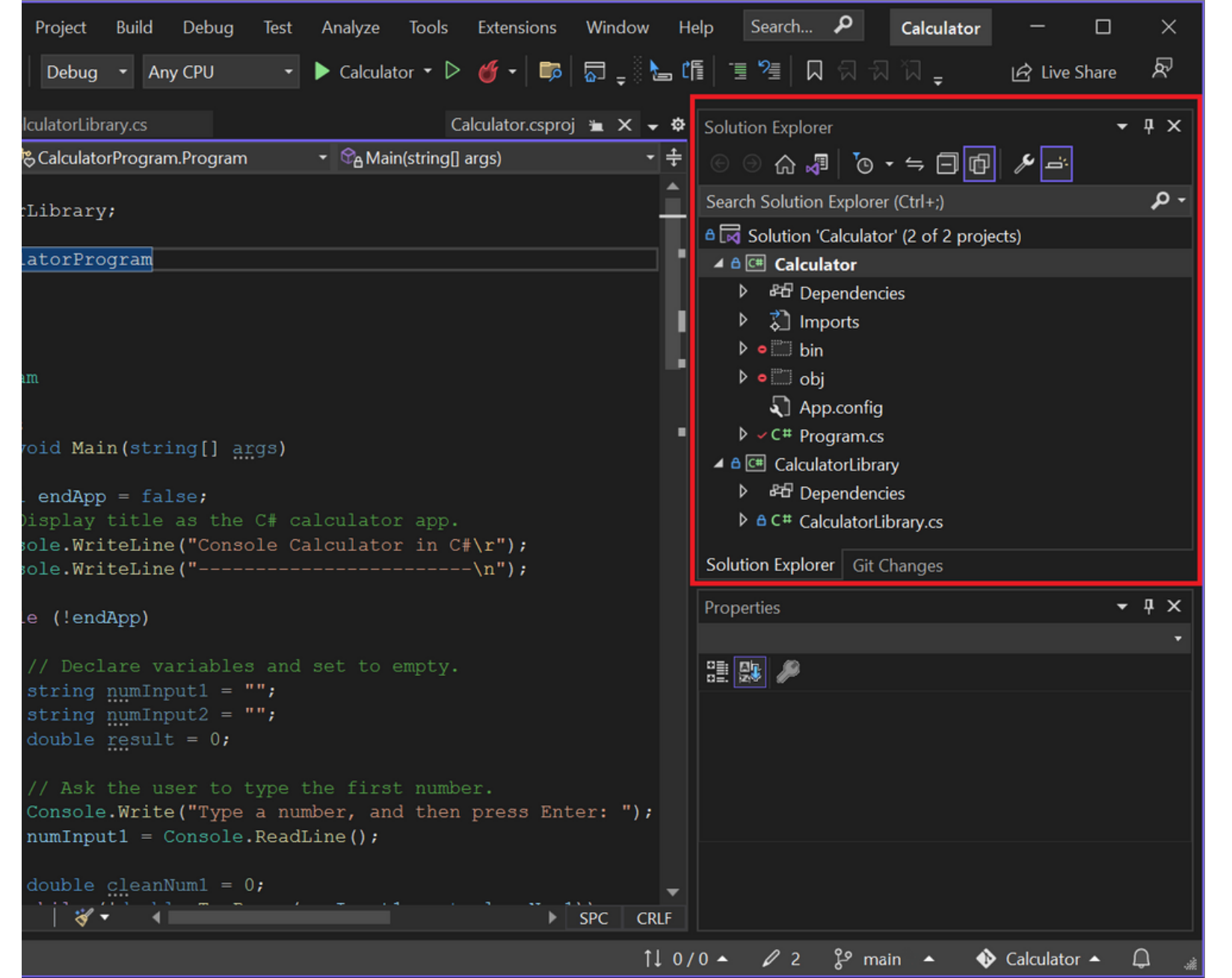
Visual Studio

Kategori: IDE

Website: <https://www.visualstudio.com/vs/>
Python araçları: Python Tools for Visual Studio

Artıları: Diğer geliştirme etkinlikleri için zaten Visual Studio'yu yüklediyseniz, PTVS eklemek daha hızlı ve kolaydır.

Eksileri: Yalnızca Python kullanmak için büyük bir indirmedir.



Python Destekli Spesifik Editörler ve IDE'ler

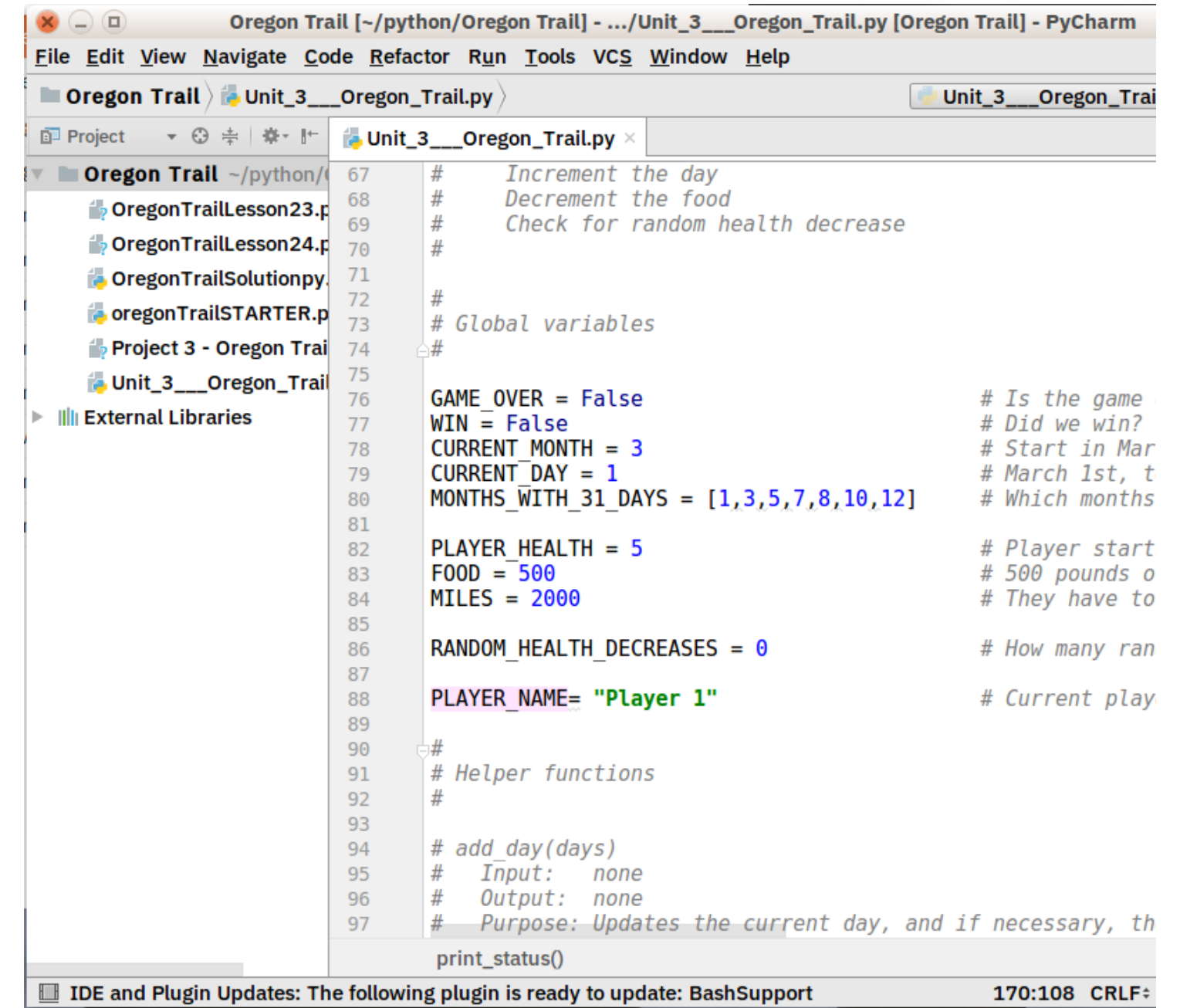
PyCharm

Kategori: IDE

Website: <https://www.jetbrains.com/pycharm/>

Artıları: Tonlarca desteğe ve destekleyici bir topluluğa sahip fiili Python IDE ortamı. Python'u kutudan çıkarır, düzenler, çalıştırır ve hata ayıklar.

Eksileri: PyCharm'ın yüklenmesi yavaş olabilir ve mevcut projeler için varsayılan ayarlarda ince ayar yapılması gerekebilir.



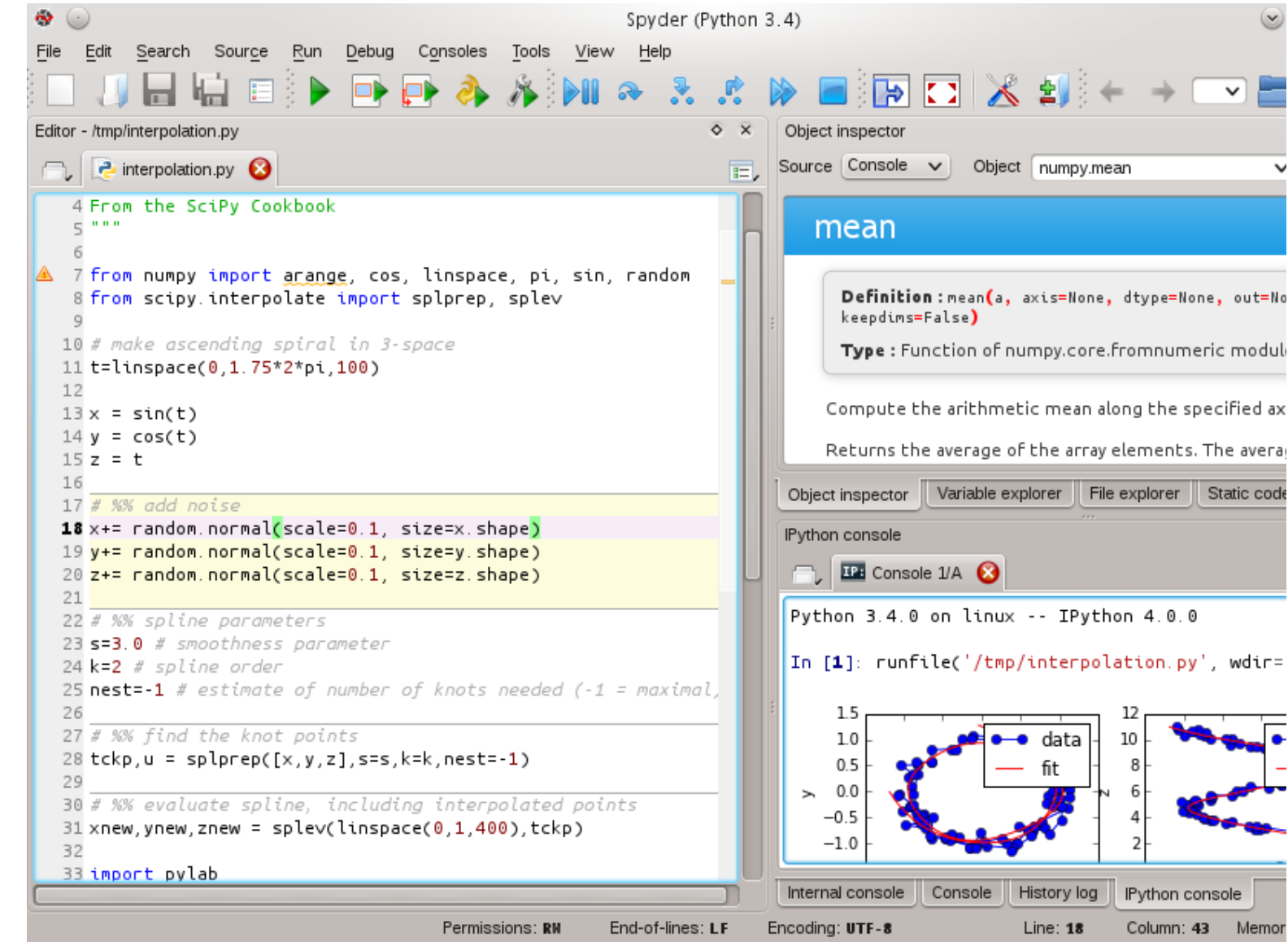
Python Destekli Spesifik Editörler ve IDE'ler

Spyder

Kategori: IDE

Website: <https://github.com/spyder-ide/spyder>

Spyder, veri bilimi iş akışları için optimize edilmiş açık kaynaklı bir Python IDE'dir. Spyder, Anaconda paket yöneticisi dağıtımına dahil olarak gelir, bu nedenle kurulumunuza bağlı olarak makinenizde zaten kurulu olabilir.



Jupyter

Kategori: IDE

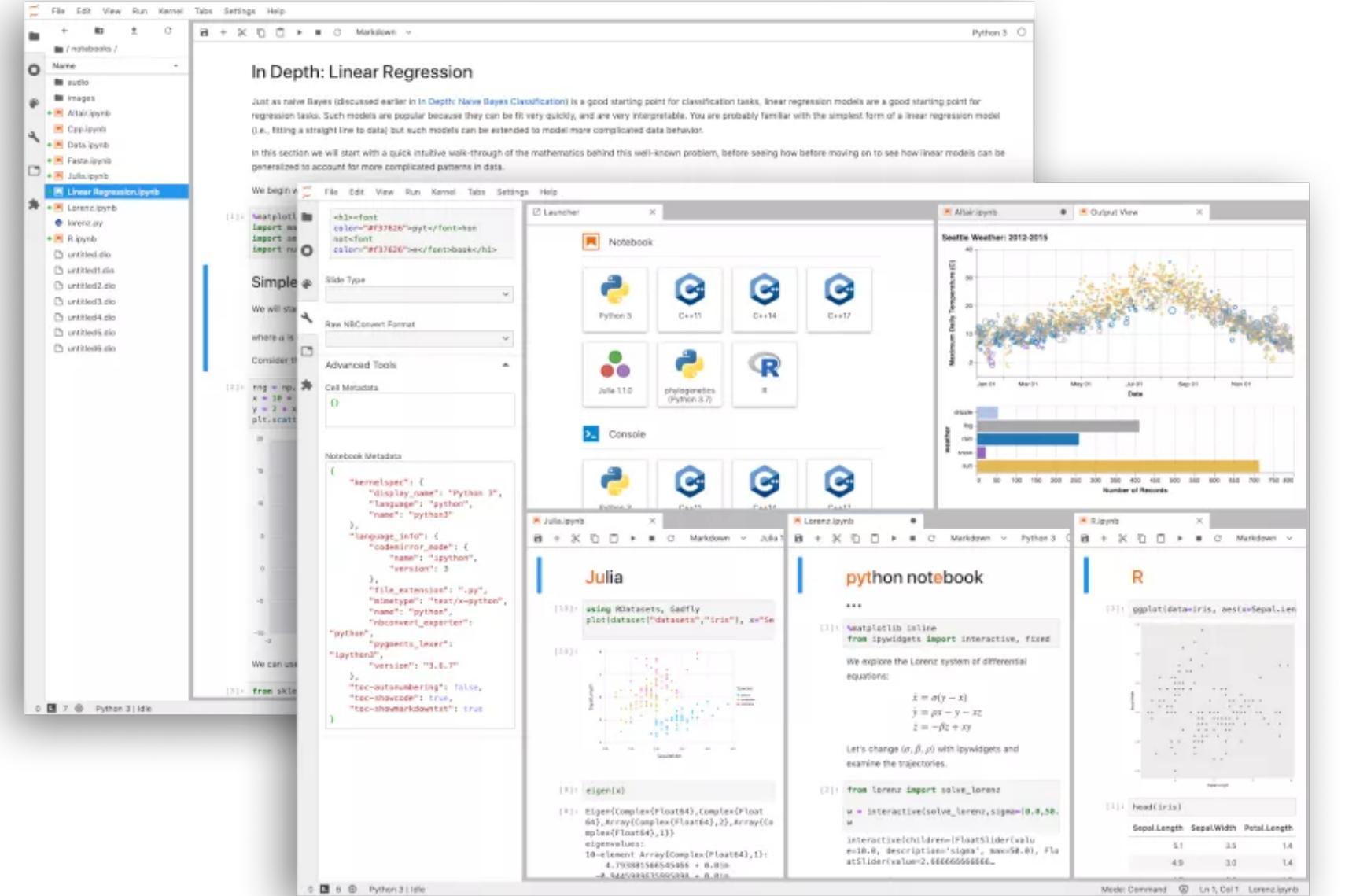
Website: **<https://jupyter.org>**

Project Jupyter, birden çok programlama dilinde etkileşimli bilgi işlem için açık kaynaklı yazılım, açık standartlar ve hizmetler geliştirme hedefleri olan bir projedir. 2014 yılında Fernando Pérez ve Brian Granger tarafından IPython'dan ayrıldı.



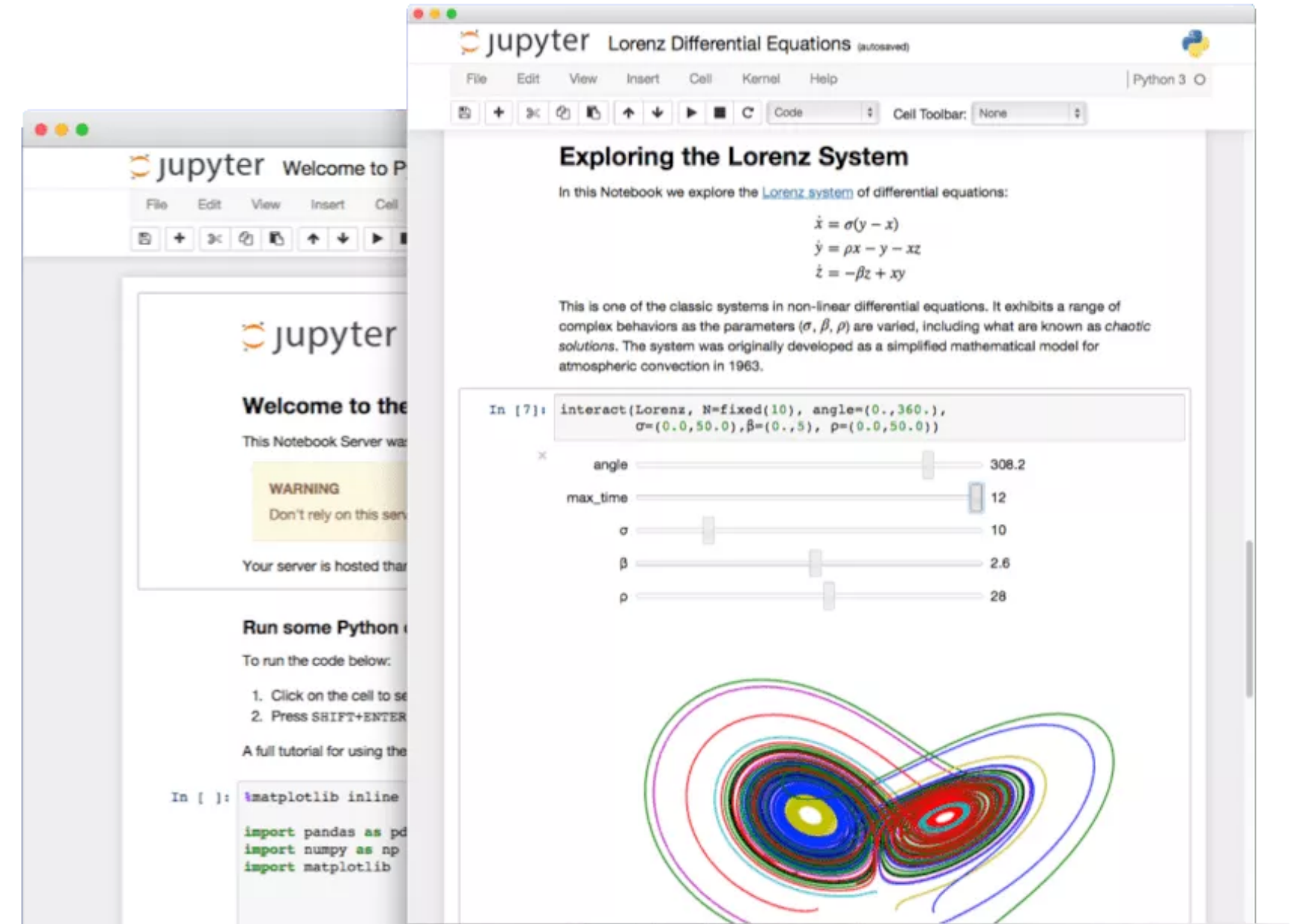
JupyterLab: Yeni Nesil Dizüstü Bilgisayar Arayüzü

JupyterLab, not defterleri, kod ve veriler için en yeni web tabanlı etkileşimli geliştirme ortamıdır. Esnek arayüzü, kullanıcıların veri bilimi, bilimsel bilgi işlem, hesaplamalı gazetecilik ve makine öğrenimindeki iş akışlarını yapılandırmasına ve düzenlemesine olanak tanır. Modüler bir tasarım, uzantıları işlevselliği genişletmeye ve zenginleştirmeye davet ediyor.



Jupyter Notebook: Klasik Notebook Arayüzü

Jupyter Notebook, hesaplama belgeleri oluşturmak ve paylaşmak için orijinal web uygulamasıdır. Basit, akıcı, belge merkezli bir deneyim sunar.



Jupyter Notebook: Klasik Notebook Arayüzü

Jupyter Notebook, veri bilimi projelerini etkileşimli olarak geliştirmek ve sunmak için inanılmaz derecede güçlü bir araçtır. Bu makale, veri bilimi projeleri için Jupyter Notebook'ların nasıl kullanılacağı ve yerel makinenizde nasıl kurulacağı konusunda size yol gösterecektir.

İlk olarak, yine de: "notebook" nedir?

Bir not defteri, **kodu ve çıktısını görselleştirmeleri, anlatı metnini, matematiksel denklemleri ve diğer zengin medyayı birleştiren tek bir belgede birleştirir**. Başka bir deyişle: kod çalıştırabileceğiniz, çıktıyı görüntüleyebileceğiniz ve ayrıca açıklamalar, formüller, çizelgeler ekleyebileceğiniz ve çalışmanızı daha şeffaf, anlaşılır, tekrarlanabilir ve paylaşılabılır hale getirebileceğiniz tek bir belgedir.

Jupyter Notebook: Klasik Notebook Arayüzü

Anaconda, sayısız kurulumu yönetme veya bağımlılıklar ve işletim sistemine özgü (okuma: Windows'a özgü) kurulum sorunları hakkında endişelenmeden, tam olarak stoklanmış bir veri bilimi atölyesi ile çalışmaya başlamanızı sağlar.

Anaconda'yı almak için basitçe:

- Python 3.8 için Anaconda'nın en son sürümünü indirin.
- İndirme sayfasındaki yürütülebilir dosyadaki talimatları izleyerek Anaconda'yı kurun. Python'un zaten kurulu olduğu daha ileri düzey bir kullanıcıysanız ve paketlerinizi manuel olarak yönetmeyi tercih ediyorsanız, sadece pip kullanabilirsiniz:

```
pip3 install jupyter
```

İlk Not Defterinizi Oluşturma

Bu bölümde, not defterlerini çalıştırmayı ve kaydetmeyi, yapılarını tanımayı ve arayüzü anlamayı öğreneceğiz. Sizi Jupyter Defterlerini kendi başınıza nasıl kullanacağınıza dair pratik bir anlayışa yönlendirecek bazı temel terminolojiyle yakınlaşacağız ve bizi örnek bir veri analizinden geçen ve burada öğrendiğimiz her şeyi hayata geçiren bir sonraki bölüme ayarlayacağız.



Files

Running

Clusters

Select items to perform actions on them.

☐ 0



☐ 3D Objects

☐ Contacts

☐ Desktop

☐ Documents

☐ Downloads

☐ Favorites

İlk Not Defterinizi Oluřturma

Windows'ta, Anaconda'nın bařlat menüñüze eklediđi kısayol aracılıđıyla Jupyter'ı alıřtırabilirsiniz; bu, varsayılan web tarayıcınızda ařađıdaki ekran görüntüsü gibi görünmesi gereken yeni bir sekme açar.



İlk Not Defterinizi Oluşturma

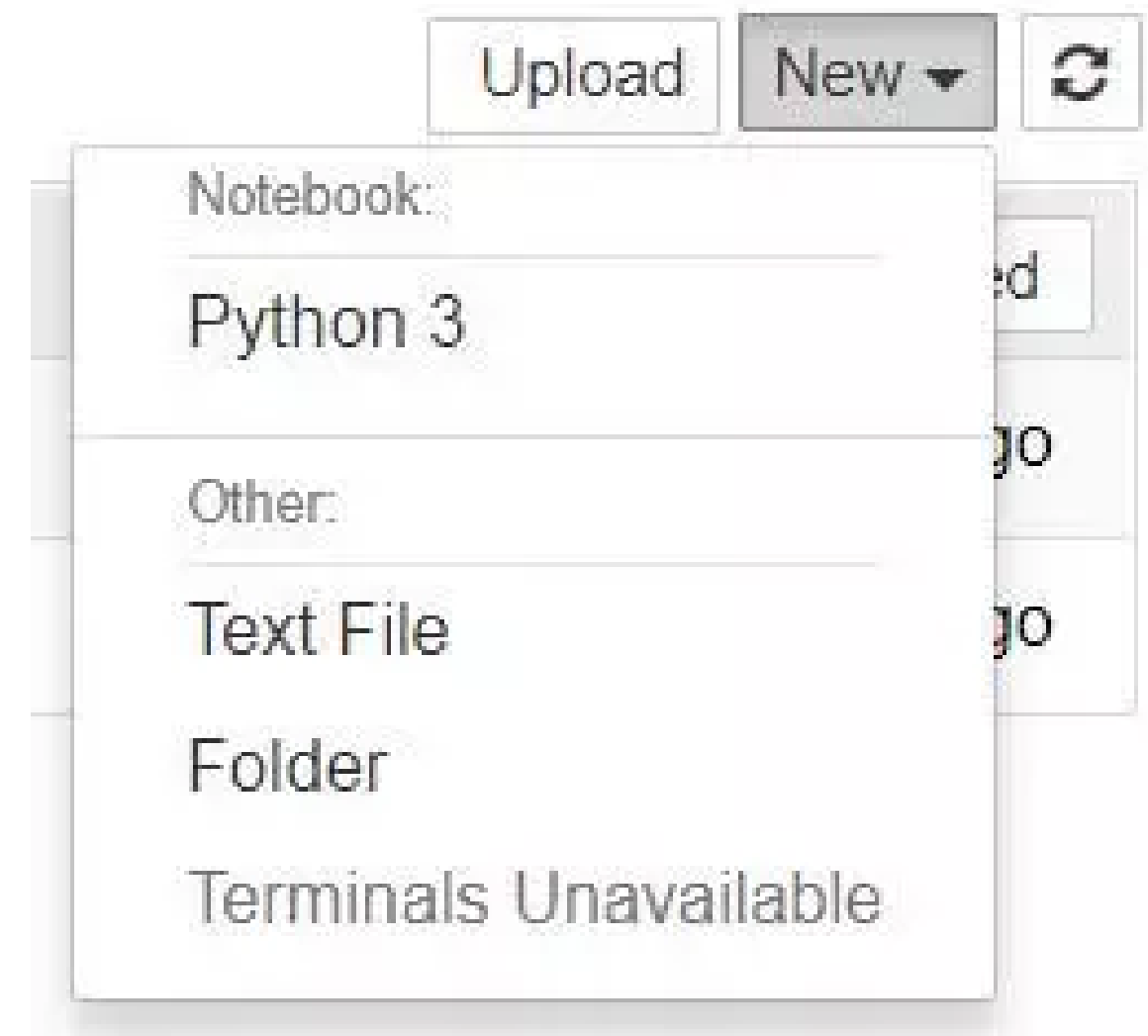
Bu henüz bir defter değil, ama panik yapmayın! Çok fazla bir şey yok. Bu, özellikle Jupyter Notebook'larınızı yönetmek için tasarlanmış Notebook Dashboard'dur. Bunu not defterlerinizi keşfetmek, düzenlemek ve oluşturmak için fırlatma rampası olarak düşünün.

Panonun size yalnızca Jupyter'ın başlangıç dizininde (yani Jupyter veya Anaconda'nın kurulu olduğu yerde) bulunan dosyalara ve alt klasörlere erişim vereceğini unutmayın. Ancak, başlangıç dizini değiştirilebilir.

Tarayıcınızda Jupyter Notebook açıkken, panonun URL'sinin <https://localhost:8888/tree> gibi bir şey olduğunu fark etmiş olabilirsiniz. Localhost bir web sitesi değildir, ancak içeriğin yerel makinenizden, yani kendi bilgisayarınızdan sunulduğunu belirtir.

İlk Not Defterinizi Oluşturma

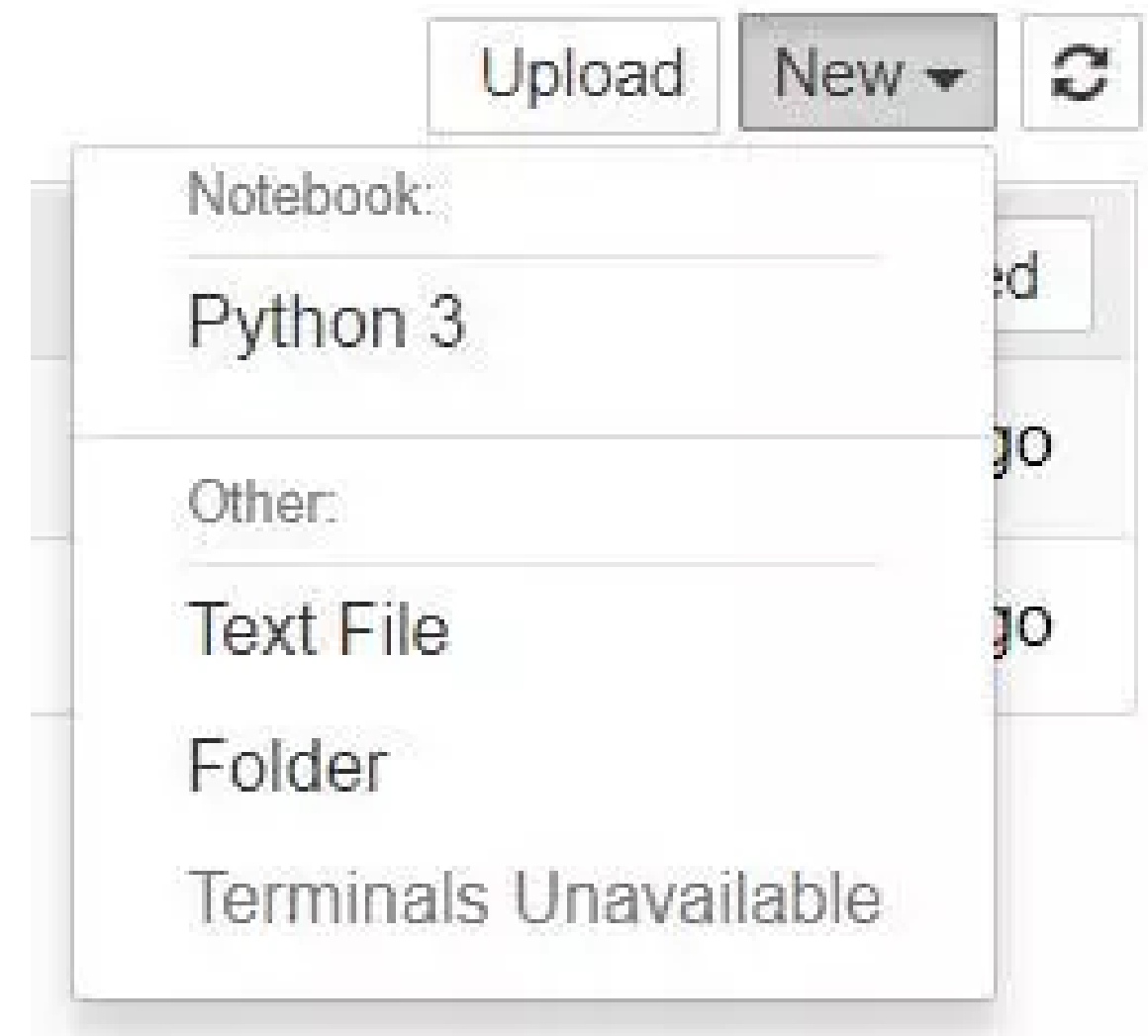
Gösterge panelinin arayüzü çoğunlukla kendi kendini açıklayıcıdır - ancak buna daha sonra kısaca geri döneceğiz. Peki ne bekliyoruz? İlk defterinizi oluşturmak istediğiniz klasöre gidin, sağ üstteki “Yeni” açılır düğmesine tıklayın ve “Python 3”ü seçin:



İlk Not Defterinizi Oluşturma

İlk Jupyter Notebook'unuz yeni sekmede açılacaktır - aynı anda birden fazla not defteri açabileceğiniz için her not defteri kendi sekmesini kullanır.

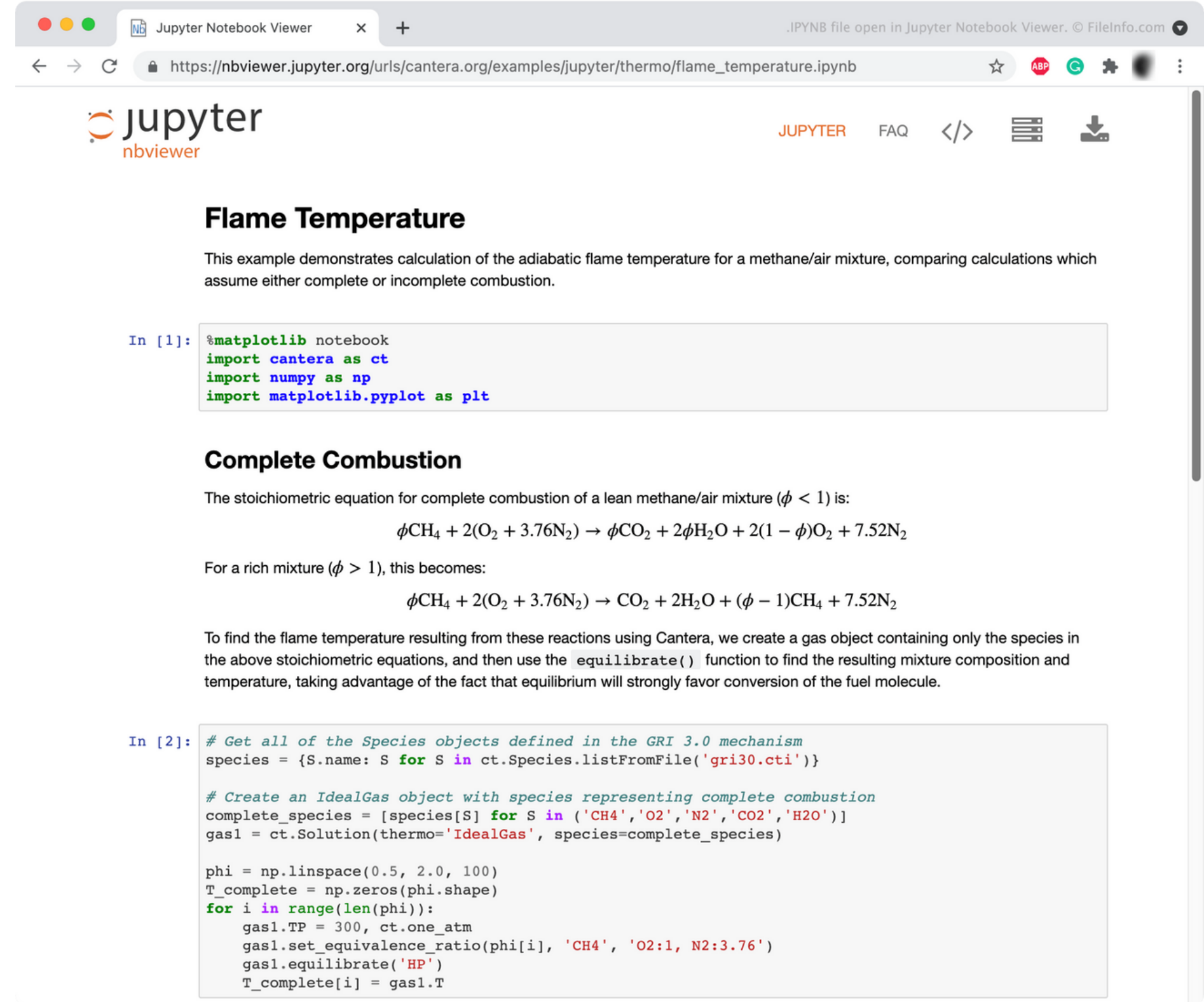
Panoya geri dönerseniz, **Untitled.ipynb** adlı yeni dosyayı göreceksiniz ve not defterinizin çalıştığını bildiren yeşil bir metin görmelisiniz.



ipynb Not Defteri Nedir?

Kısa cevap: her .ipynb dosyası bir not defteridir, bu nedenle her yeni not defteri oluşturduğunuzda, yeni bir .ipynb dosyası oluşturulur.

Daha uzun yanıt: Her .ipynb dosyası, not defterinizin içeriğini JSON adlı bir biçimde açıklayan bir metin dosyasıdır. Metin dizelerine dönüştürülmüş resim ekleri dahil olmak üzere her hücre ve içeriği, bazı meta verilerle birlikte burada listelenir.

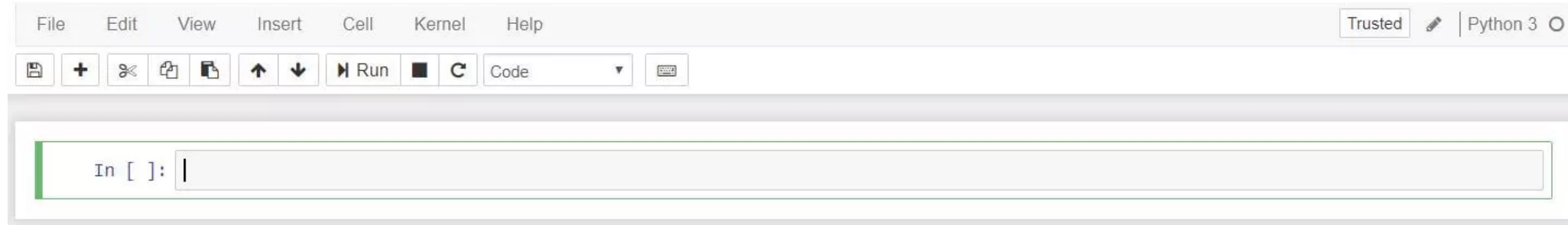


The screenshot shows a web browser window with the Jupyter Notebook Viewer interface. The address bar displays the URL: https://nbviewer.jupyter.org/urls/cantera.org/examples/jupyter/thermo/flame_temperature.ipynb. The notebook title is "Flame Temperature". Below the title, a description states: "This example demonstrates calculation of the adiabatic flame temperature for a methane/air mixture, comparing calculations which assume either complete or incomplete combustion." The notebook content is divided into two sections: "Complete Combustion" and "Incomplete Combustion". The "Complete Combustion" section includes a stoichiometric equation for a lean mixture ($\phi < 1$):
$$\phi\text{CH}_4 + 2(\text{O}_2 + 3.76\text{N}_2) \rightarrow \phi\text{CO}_2 + 2\phi\text{H}_2\text{O} + 2(1 - \phi)\text{O}_2 + 7.52\text{N}_2$$
 and a similar equation for a rich mixture ($\phi > 1$):
$$\phi\text{CH}_4 + 2(\text{O}_2 + 3.76\text{N}_2) \rightarrow \text{CO}_2 + 2\text{H}_2\text{O} + (\phi - 1)\text{CH}_4 + 7.52\text{N}_2$$
. The "Incomplete Combustion" section explains how to find the flame temperature using Cantera. The notebook contains two code cells. The first cell (In [1]) imports the necessary libraries:

```
In [1]: %matplotlib notebook
import cantera as ct
import numpy as np
import matplotlib.pyplot as plt
```

 The second cell (In [2]) contains a more complex script that defines species, creates an IdealGas object, and calculates the adiabatic flame temperature for a range of equivalence ratios (ϕ) from 0.5 to 2.0. The script uses Cantera's `equilibrate()` function to find the resulting mixture composition and temperature.

ipynb Not Defteri Nedir?



Artık önünüzde açık bir not defteriniz olduğuna göre, arayüzü umarım tamamen yabancı görünmeyecektir. Sonuçta, Jupyter aslında sadece gelişmiş bir kelime işlemcidir.

Neden etrafa bir göz atmıyorsunuz? Bir fikir edinmek için menülere göz atın, özellikle klavye simgesi (veya **Ctrl + Shift + P**) içeren küçük düğme olan komut paletindeki komut listesini aşağı kaydırmak için birkaç dakikanızı ayırın.

Jupyter Notebook

Dikkat etmeniz gereken, muhtemelen sizin için yeni olan, oldukça belirgin iki terim vardır: **hücreler** ve **çekirdekler**, hem Jupyter'i anlamamanın hem de onu bir kelime işlemciden daha fazlasını yapan şeyin anahtarıdır. Neyse ki, bu kavramları anlamak zor değil.

- Çekirdek, bir not defteri belgesinde bulunan kodu yürüten bir "hesaplama motorudur".
- Hücre, not defterinde görüntülenecek metin veya not defterinin çekirdeği tarafından yürütülecek kod için bir kapsayıcıdır.

Jupyter Notebook - Hücreler

Biraz sonra çekirdeklere döneceğiz, ama önce hücrelerle uğraşalım. Hücreler bir defterin gövdesini oluşturur. Yukarıdaki bölümdeki yeni bir not defterinin ekran görüntüsünde, yeşil çerçeveli kutu boş bir hücredir. Ele alacağımız iki ana hücre tipi vardır:

- Bir kod hücresi, çekirdekte yürütülecek kodu içerir. Kod çalıştırıldığında, not defteri çıktıyı, onu oluşturan kod hücresinin altında görüntüler.
- Bir Markdown hücresi, **Markdown** kullanılarak biçimlendirilmiş metin içerir ve Markdown hücresi çalıştırıldığında çıktısını yerinde görüntüler.

Jupyter Notebook - Hücreler

Yeni bir not defterindeki ilk hücre her zaman bir kod hücresidir.

- Klasik bir merhaba dünya örneği ile test edelim: Hücreye `print('Hello World!')` yazın ve yukarıdaki araç çubuğundaki çalıştır düğmesini Not Defteri Çalıştır Düğmesi'ni tıklayın veya Ctrl + Enter tuşlarına basın.



Markdown

Markdown, düz metni biçimlendirmek için hafif, öğrenmesi kolay bir biçimlendirme dilidir. Sözdizimi, HTML etiketleriyle bire bir yazışmaya sahiptir, bu nedenle burada bazı ön bilgiler yardımcı olabilir, ancak kesinlikle bir ön koşul değildir.

- Bu makalenin bir Jupyter not defterinde yazıldığını unutmayın, bu nedenle şimdiye kadar gördüğünüz tüm anlatım metinleri ve görseller Markdown'da yazılmıştır. Temel bilgileri hızlı bir örnekle ele alalım:

```
# This is a level 1 heading
```

```
## This is a level 2 heading
```

```
This is some plain text that forms a paragraph. Add emphasis via bold and bold, or italic ar
```

Markdown

Paragraphs must be separated by an empty line.

- * Sometimes we want to include lists.
- * Which can be bulleted using asterisks.

1. Lists can also be numbered.
2. If we want an ordered list.

[It is possible to include hyperlinks](<https://www.example.com>)

Inline code uses single backticks: `foo()`, and code blocks use triple backticks:

```
```\nbar()\n```
```

# Notebook

Her defterin arkasında bir çekirdek çalışır. Bir kod hücrecini çalıştırdığınızda, bu kod çekirdek içinde yürütülür. Herhangi bir çıktı, görüntülenecek hücreye geri döndürülür. Çekirdeğin durumu zamanla ve hücreler arasında devam eder - tek tek hücrelerle değil, bir bütün olarak belgeyle ilgilidir.

Örneğin, bir hücrede kitaplıkları içe aktarır veya değişkenler bildirirseniz, bunlar başka bir hücrede kullanılabilir olacaktır. Bunu hissetmek için deneyelim. İlk önce bir Python paketini içe aktaracağız ve bir fonksiyon tanımlayacağız:



# Notebook

```
import numpy as np
def square(x):
 return x * x
```

```
x = np.random.randint(1, 10)
y = square(x)
print('%d squared is %d' % (x, y))
```

Yukarıdaki hücreyi çalıştırdıktan sonra, başka herhangi bir hücrede np ve square'e başvurabiliriz.

Bu, defterinizdeki hücrelerin sırasına bakılmaksızın çalışacaktır. Bir hücre çalıştırıldığı sürece, bildirdiğiniz tüm değişkenler veya içe aktardığınız kitaplıklar diğer hücrelerde kullanılabilir olacaktır.

# Python Sınıflar

Sayılar, diziler ve listeler gibi ilkel veri yapıları, sırasıyla bir elmanın maliyeti, bir şiirin adı veya en sevdiğiniz renkler gibi basit bilgi parçalarını temsil edecek şekilde tasarlanmıştır. Ya daha karmaşık bir şeyi temsil etmek istiyorsanız?

Örneğin, bir kuruluştaki çalışanları izlemek istediğinizi varsayalım. Her çalışanın adı, yaşı, pozisyonu ve çalışmaya başladığı yıl gibi bazı temel bilgileri saklamanız gerekir.

# Python Sınıflar

Bunu yapmanın bir yolu, her çalışanı bir liste halinde temsil etmektir: Bu yaklaşımla ilgili bir takım sorunlar var.

Python

```
kirk = ["James Kirk", 34, "Captain", 2265]
spock = ["Spock", 35, "Science Officer", 2254]
mccoy = ["Leonard McCoy", "Chief Medical Officer", 2266]
```

İlk olarak, daha büyük kod dosyalarının yönetilmesini zorlaştırabilir. Kirk[0]'a kirk listesinin bildirildiği yerden birkaç satır ötede başvurursanız, indeks 0'a sahip öğenin çalışanın adı olduğunu hatırlayacak mısınız?

İkincisi, listede her çalışan aynı sayıda öğeye sahip değilse, hatalara neden olabilir. Yukarıdaki mccoy listesinde yaş eksiktir, bu nedenle mccoy[1], Dr. McCoy'un yaşı yerine "Baş Sağlık Görevlisi" değerini döndürür.

# Python Sınıflar

Python'daki hemen hemen her şey, özellikleri ve yöntemleri ile bir nesnedir.

Bir Sınıf, bir nesne oluşturucu veya nesneler oluşturmak için bir "plan" gibidir. Bir sınıf oluşturmak için **class** anahtar sözcüğünü kullanın:

```
class MyClass:
 x = 5
```

```
p1 = MyClass()
print(p1.x)
```

# Python Sınıflar

Python'daki hemen hemen her şey, özellikleri ve yöntemleri ile bir nesnedir.

Bir Sınıf, bir nesne oluşturucu veya nesneler oluşturmak için bir "plan" gibidir.  
Bir sınıf oluşturmak için **class** anahtar sözcüğünü kullanın:

```
class MyClass:
 x = 5
```

```
p1 = MyClass()
print(p1.x)
```

# Referanslar

- Sams Teach Yourself Object Oriented Programming in 21 Days, Second Edition
  - Mesleki ve Teknik A. L., Bilişim Teknolojileri, Nesne Tabanlı Programlama
  - Nesne Tabanlı Programlama I, Kitap
  - Python programlama, Wikipedia
  - W3Schools, Tutorial
-



# İletişim Bilgileri

---

ÖĞR. GÖR. BUSE YAREN TEKİN



bytekin@kastamonu.edu.tr

