

Python telepítési segédlet macOS-hez

Sykora Henrik*, Tóth Gergely, Csuzdi Domonkos
BME-GPK Műszaki Mechanikai Tanszék

2020. február 18.

1. Bevezető

1.1. A Python nyelv

A Python egy nyílt forráskódú, általános célú, *interpretált, objektum orientált* programozási nyelv. 2020-ra a legelterjedtebben alkalmazott általános célú programozási nyelvvé nőtte ki magát, széles körben használják informatikai (pl: YouTube, Instagram, Spotify web-backend), tudományos és matematikai célokra (pl: Datamining, AI development, Machine Learning). Számos előnye van a mérnöki alkalmazásokban is, rengeteg ingyenesen elérhető függvénykönyvtár készült hozzá, néhány példa:

- `numpy`: numerikus számítások, lineáris algebra, numerikus integrálás, stb...
- `sympy`: szimbolikus számítások (computer algebra)
- `PIL`: képfeldolgozás (*Python Imaging Library*)
- `matplotlib`: függvény kirajzolás, eredmények megjelenítése
- `pandas`: data mining, adatfeldolgozás
- `sklearn`: machine learning
- ...

További előnye, hogy a Python nyelv törekszik a tömörségre, olvashatóságra és átláthatóságra, így az alapok órák alatt elsajátíthatóak (ellenben pl. a C, C++, Java nyelvekkel). A nyílt forráskódnak köszönhetően ingyenesen használható, minden ismertebb platformon elérhető (ellenben pl. a Matlabbal és Mathematicával), ezért a Python-ban írt kód mindenhol könnyen futtatható.

1.2. Fejlesztői eszközök

A Python nyelv fejlesztői által hivatalosan készített, úgynevezett 'core' (lásd még: interpreter, mag, fordító) funkciói erősen korlátozottak. A [Python.org](https://python.org) oldalról letölthető gyári csomag segítségével, ugyan futtathatjuk a programunk kódját egy console applikáción keresztül, de ebben a formában semmilyen pár sornál hosszabb kódot nem praktikus írni. Ami teljessé teszi a Python-nal történő munkánkat, azok a felhasználó közösség által fejlesztett IDE-k és Package managerek.

1.3. Package manager-ek

A Python nyelv fő előnye az alternatíváival szemben a moduláris jellegében rejlik. A különböző alkalmazásokhoz, különböző package-k (függvénykönyvtárak) állnak rendelkezésünkre, amiket valamilyen módon rendszerezniük kell. A jelen szöveg írásakor a legelterjedtebb package managerek a következők:

- `PIP`: A Python gyári package kezelő modulja. Sok funkcióval rendelkezik, de kezdők számára sok utánajárást igényelhet a használata (Terminál ablakból lehet a PIP-el kommunikálni).
- `Virtualenv`: Lehetővé teszi a virtuális környezetek könnyű kezelését, amire szükség lehet, ha ugyan azon packagek több fajta verzióját akarjuk felváltva használni. Elsősorban haladó fejlesztőknek ajánlott.

*Észrevételeiket vagy a talált hibákat kérem, hogy a sykora@mm.bme.hu e-mail címen jelezzék

- Anaconda: A mérnöki és data-science igényekre kifejlesztett manager az Anaconda. Grafikus felületével egyszerűen kezelhető, és gyárilag telepíti a legfontosabb package-eket. A segédlet további részében az Anaconda használata lesz részletezve.

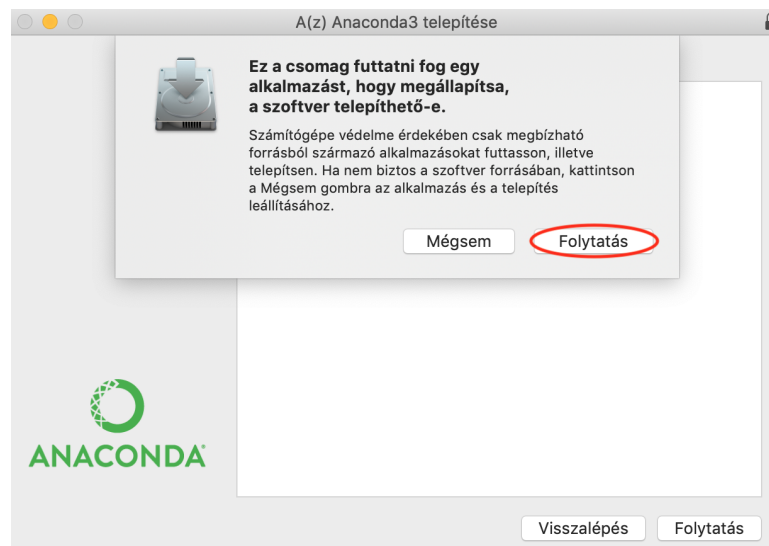
1.4. IDE-k

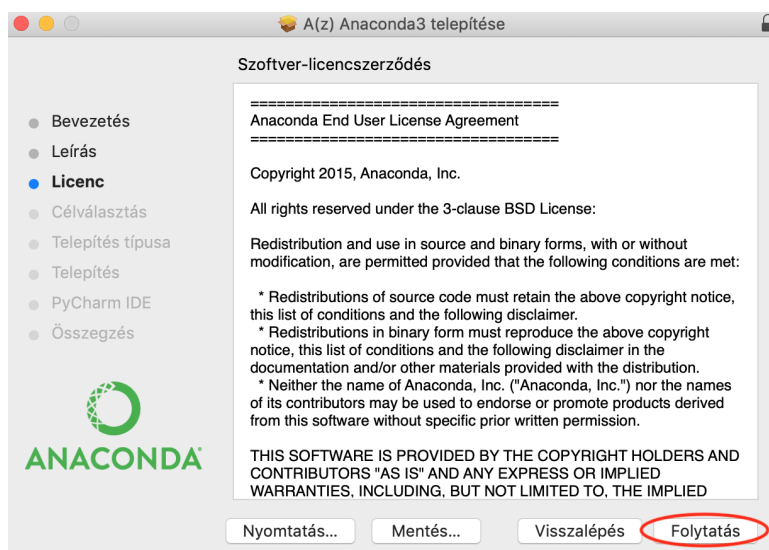
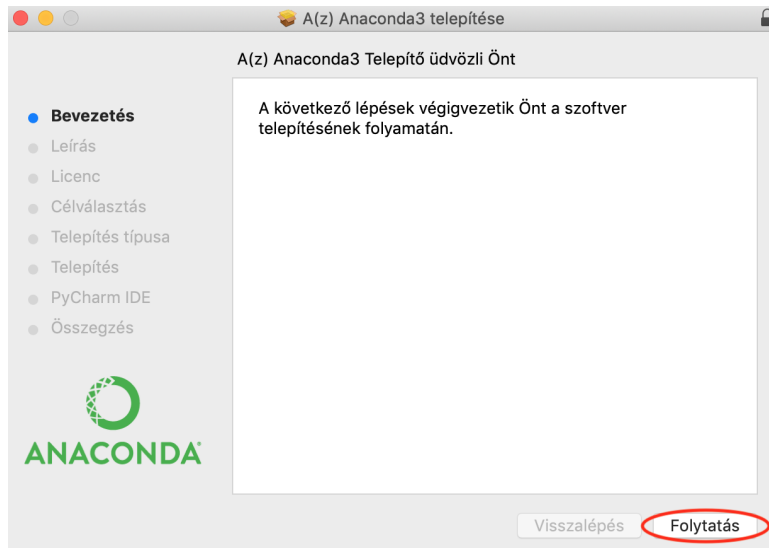
Az IDE (Integrated Development Environment) programok egy felületet teremtenek a felhasználó számára, amivel kommunikálhat a programkódot értelmező core-al. Számos segítő funkciót nyújtanak a kód debuggolásában, átláthatóbbá tételében, optimalizálásában. Egyes IDE-k lehetővé tesznek interaktív notebook és interaktív ábra készítést is. A jelen szöveg írásakor a legelterjedtebb IDE-k a következők:

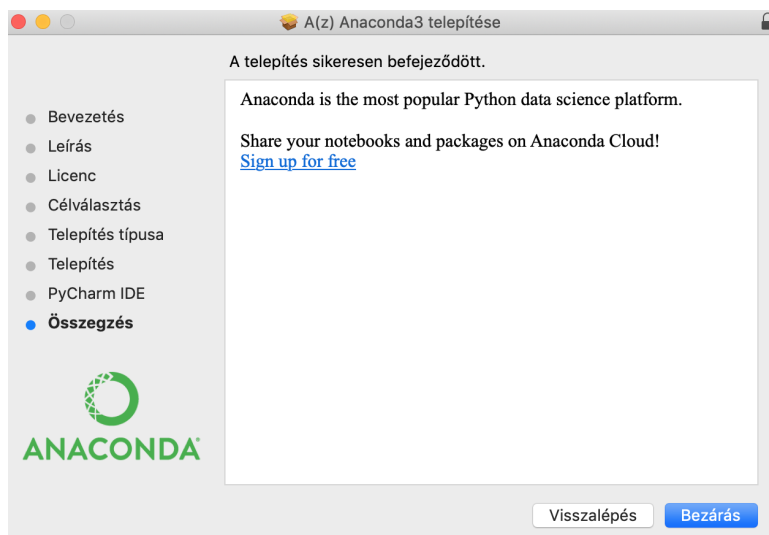
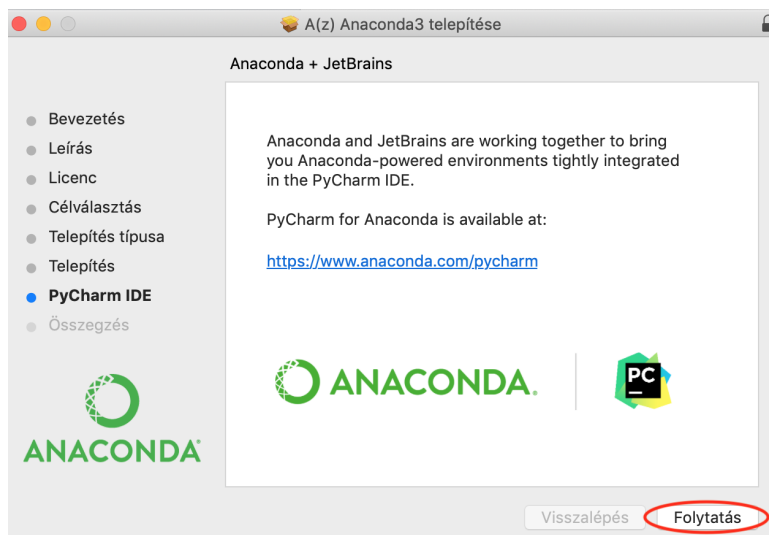
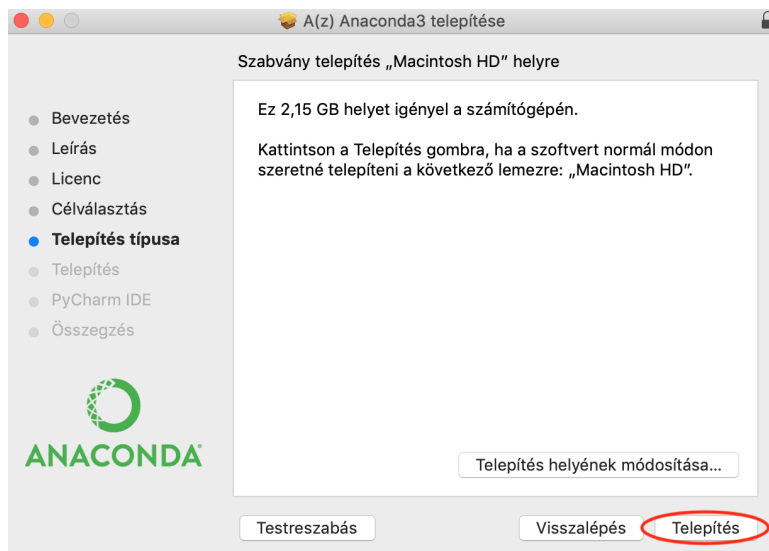
- IDLE: A *Python* gyári kód szerkesztő IDE-je. Minimalista mind a desing-t mind a funkcióit tekintve.
- Spyder: Tudományos és programfejlesztői munkára egyaránt alkalmas, ingyenes kódszerkesztő, debuggolást segítő funkciókkal.
- VS Code: A Microsoft Visual Studio mintáit követő, ingyenesen elérhető kódszerkesztő, kiterjedt fejlesztői eszközökkel. Nagyobb projekteknél értékelhetőek igazán a VS Code nyújtotta előnyök.
- Jupyter: A *Python*-t interaktív képességekkel felruházó IDE. A Jupyter Notebook-ban lehetőség van a programunkat több cellába rendezni, és a feladathoz kötődő formázott szöveget közvetlenül a kódunk cellái közé illeszteni, illetve interaktív ábrákat, diagrammokat készíteni. Ezeknek a funkcióknak köszönhetően elterjedten használják tudományos és műszaki feladatokban is.
- Jupyterlab: A Jupyter fejlesztőinek egy feltörekvő kezdeményezése, ami még több grafikus és interaktív lehetőséget biztosít majd a *Python* programozóknak. (Nem titkoltan a *Matlab* és a *Wolfram Mathematica* programokkal való verseny inspirálta)

2. Telepítés lépései

Töltsük le a következő helyről, a számunkra megfelelő verzióját az Anaconda csomagnak: [Anaconda 2019.10 for macOS Installer - Python 3.7 - 64-Bit Graphical Installer \(654 MB\)](#). Indítsuk el a telepítőt:





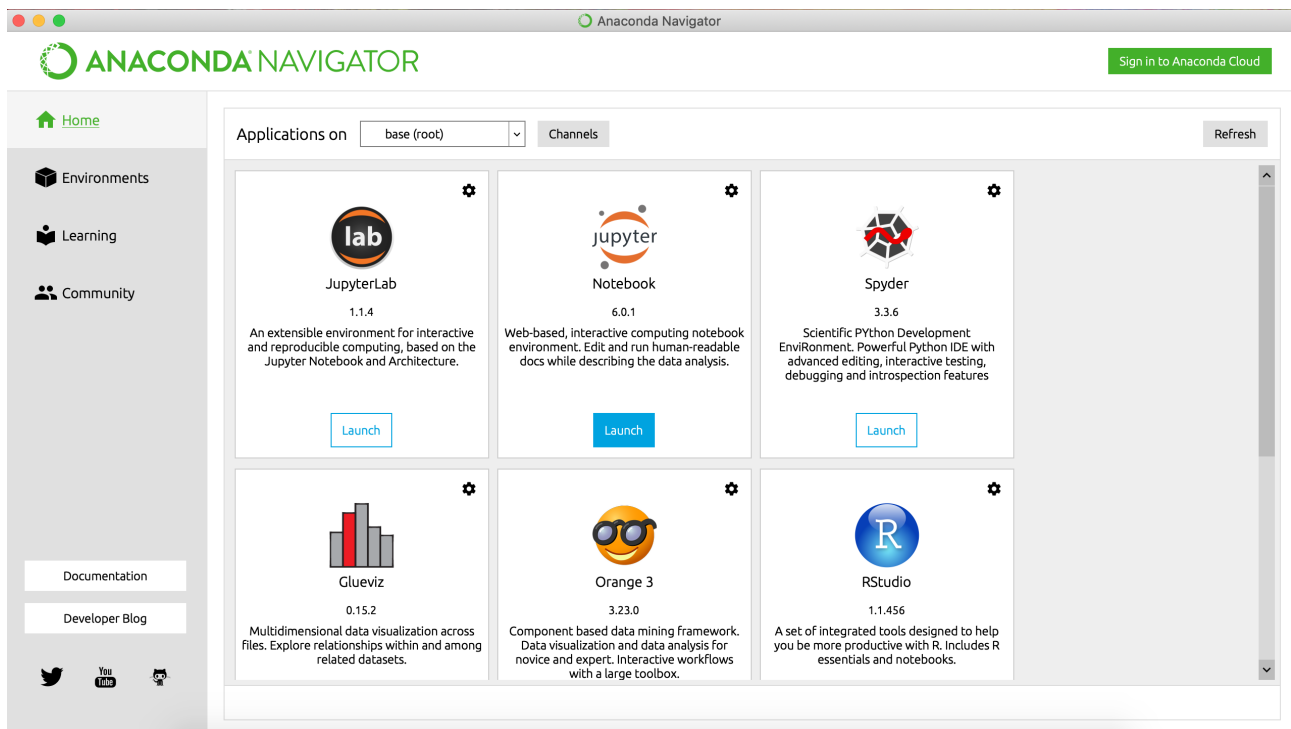


3. Python futtatása

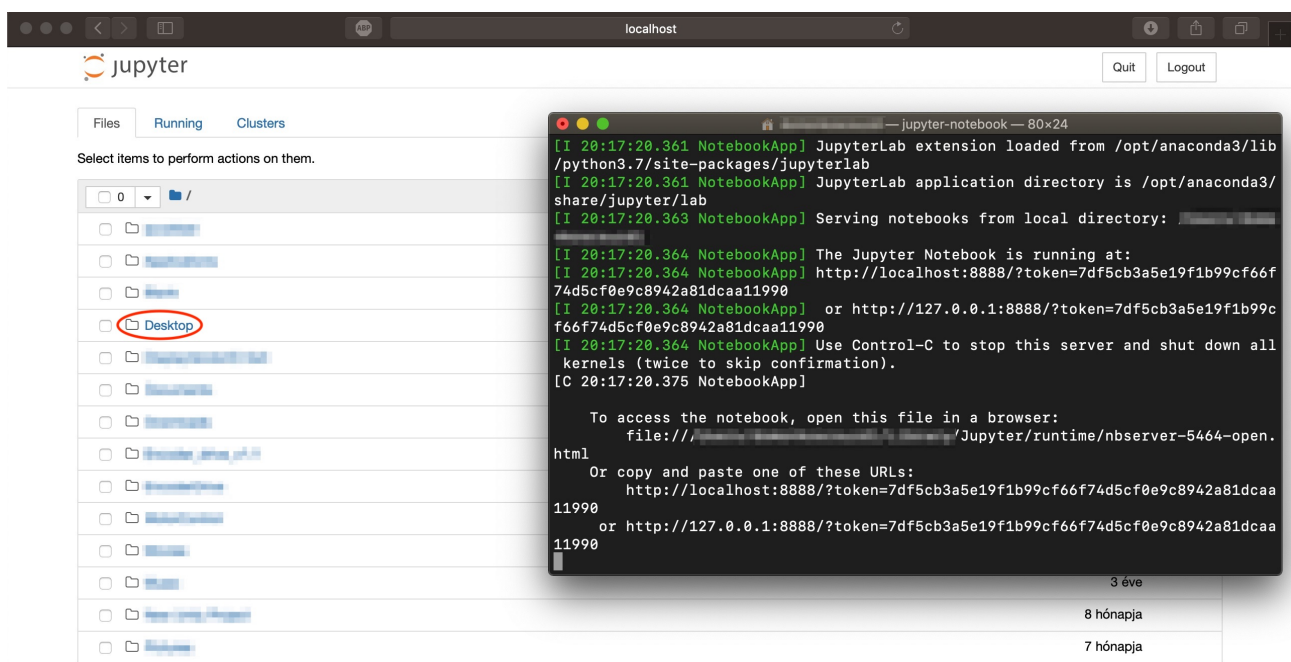
Most hogy megvagyunk az Anaconda és a Python telepítésével, több módon is elindíthatjuk a Jupyter Notebook szerkesztőt. (A Navigator-ból elérhető több különböző IDE is, amit érdekességképp ki lehet próbálni.)

3.1. Python futtatása Anaconda Navigator-on keresztül

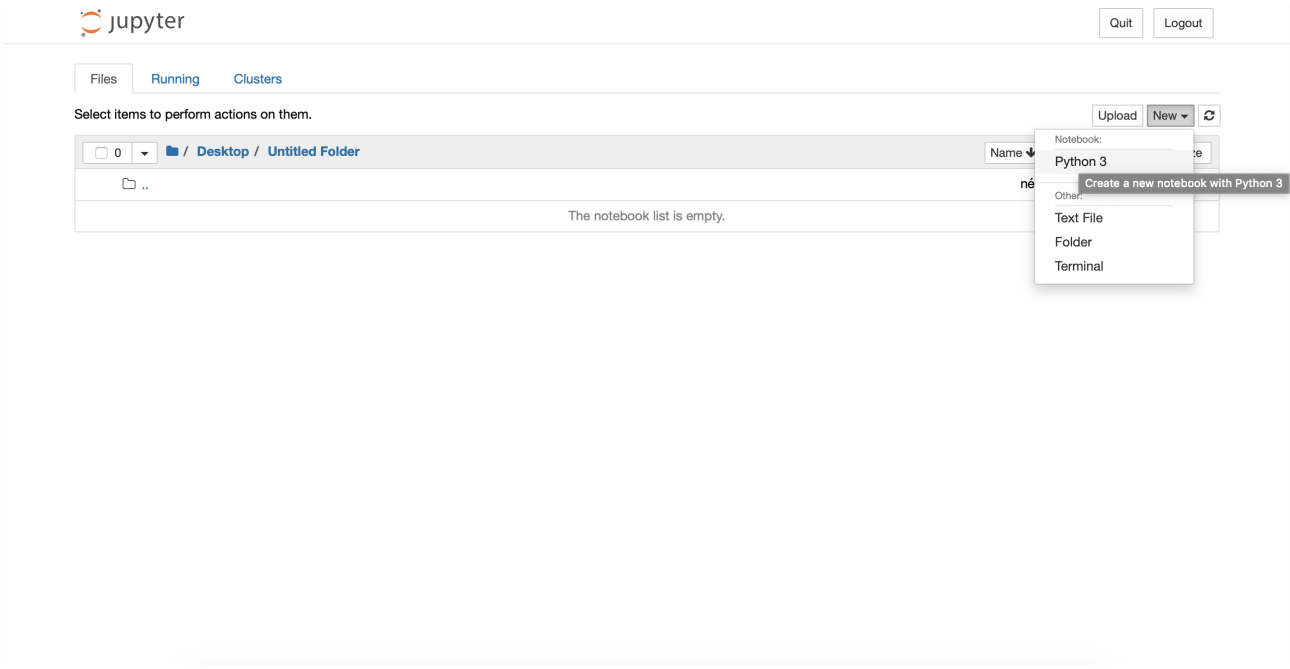
A telepített programok közül futtassuk az Anaconda Navigator-t, majd kattintsunk a Jupyter Notebook alatti Launch ikonra.



Amint elindítjuk a Jupyter Notebookot, megnyílik egy terminál, valamint egy böngésző ablakban a Jupyter kezelőfelülete, ahol a továbbiakban dolgozni fogunk. **Fontos, hogy a terminált *ne* zárjuk be, nyugodtan tegyük le a tálcára.**

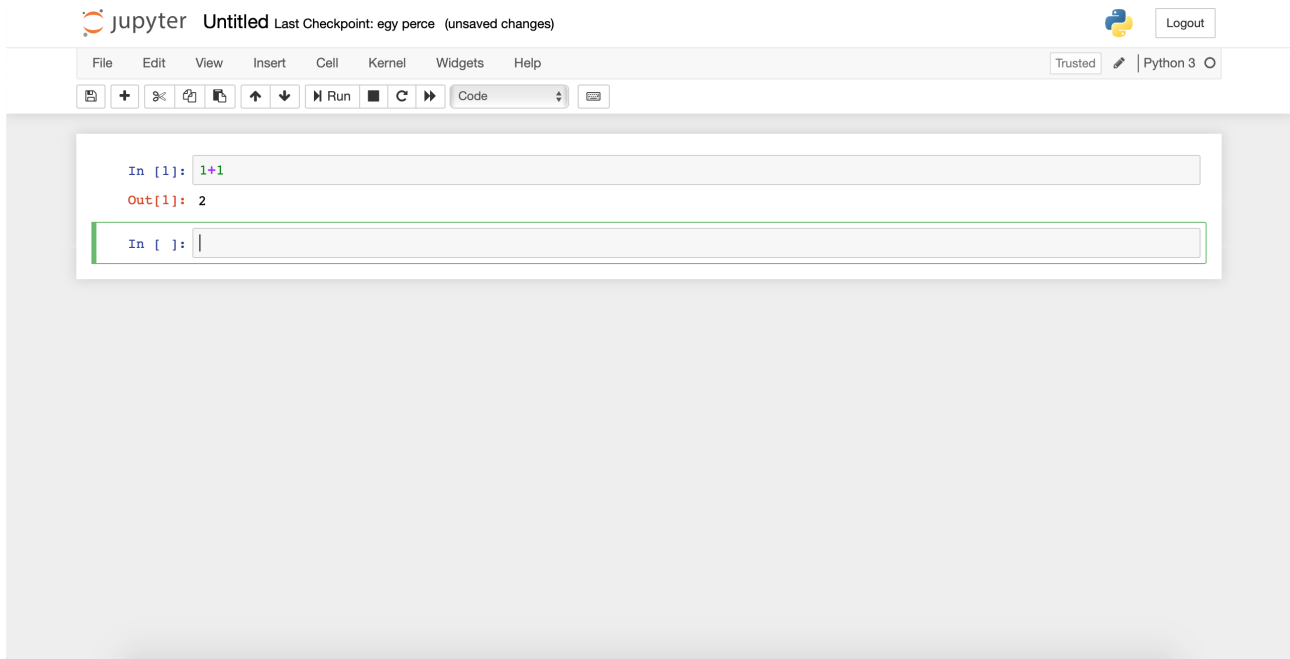
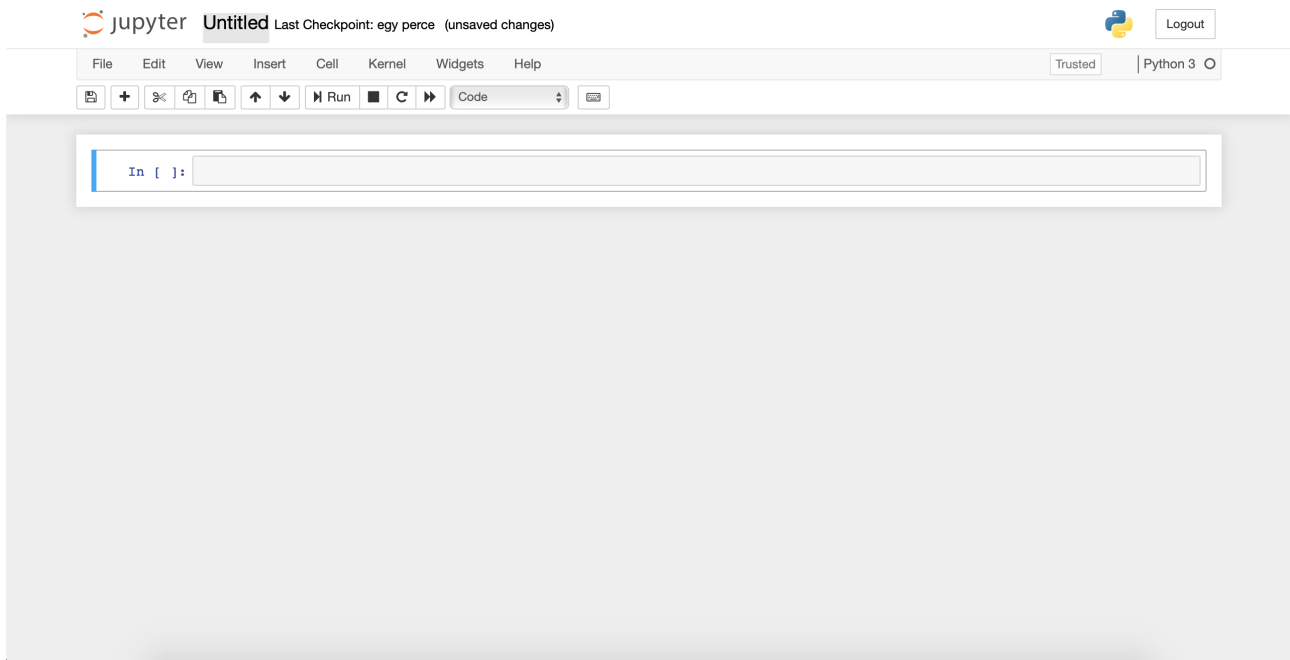


Hozzunk létre egy új mappát, pl. az asztalon, melyben dolgozni fogunk. Navigáljunk ide el az által, hogy a megjelenő listában a *Desktop* mappa *névére*, majd az ezt követően megjelenő listában az elkészített mappára kattintunk. Itt létrehozunk egy új, *Python 3*-as notebook-ot.



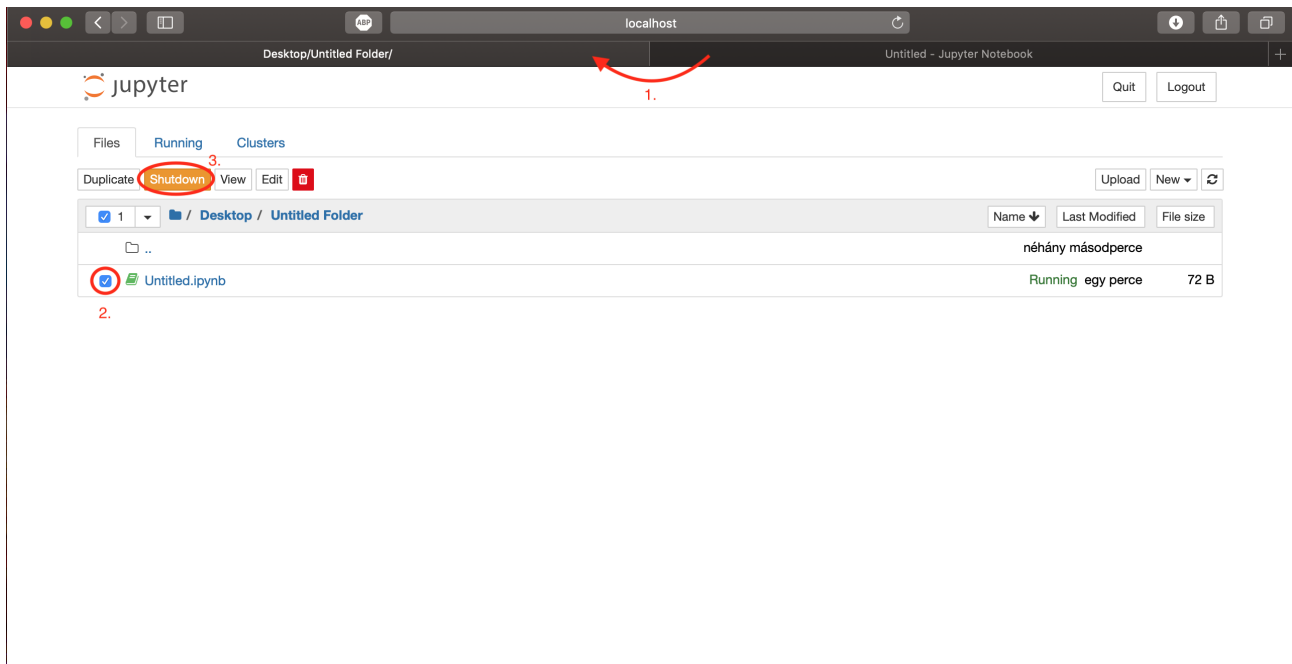
4. Parancsok és navigálás a Jupyter felületen

A Jupyter biztosít számunkra egy böngészőt, amivel böngészhetünk, létrehozhatunk, módosíthatunk és törölhetünk fileokat. A notebookban a cellák *shift+enter* vagy *ctrl+enter* segítségével futtathatóak, és ha kéken van kijelölve a cella (*Command Mode* - *esc* billentyű lenyomásával elérhető), akkor a *shift+L* segítségével bekapcsolható a sorok számozása. Szerkesztő módban (*Editor Mode* - cellát kijelölve *enter* lenyomása) cellákban akár egyszerre több kurzor is helyezhető a *cmd-balklíkk* segítségével, vagy téglalapszerűen is kijelölhető a szöveg az *alt* nyomvatartása mellett. A jupyter környezet help-je a "h" megnyomásával előhívható *Command Mode*-ban. A notebook átnevezhető a felső részben lévő „Untitled”-re kattintva.



4.1. A Jupyter leállítása

A futó notebook (zöld könyvvel jelezve) a Jupyter menüjéből (a böngészőben a másik fülre átváltva) azután állítható le, hogy a mellette lévő jelölőnégyzetet bepipáljuk. Ezt követően van lehetőség leállítani a *Shutdown* gombbal, majd zárjuk be a notebookhoz tartozó böngészőablakot is (a felugró figyelmeztetés ellenére el lehet hagyni az oldalt). Ez után szükséges a Jupyter szervert is leállítani (amennyiben nem tervezzük tovább dolgozni), melyet a jobb felső sarokban lévő *Quit* gombbal tehetünk meg. Mostmár a terminál ablak is bezárható. (Alternatív szerver leállítási módszer, ha a terminál ablakban kétszer megnyomjuk a `ctrl+c` billentyűkombinációt, melynek következtében bezárható a Jupyter böngészőablaka is.)



5. *Python* futtatása Jupyter Notebook-ban, terminálon keresztül

A Jupyter Notebook futtatható olyan formában is, hogy azt ne Anaconda Navigator-on keresztül nyissuk meg. Ehhez nyomjuk meg a *cmd+space* billentyűkombinációt, írjuk be a **Terminal** parancsot, ezzel megnyitva azt. A Jupyter Notebook a `jupyter notebook` paranccsal indítható, *enter* lenyomása után.

