Workshop

Workshop-Themen

- Python-Entwicklungsumgebungen:
  - Spyder
  - Jupyter
- Branchensoftware mit Python-Schnittstelle:
  - Blender
  - optional: FreeCAD
- CAMPLA/Lernstick

Spyder

Python-Entwicklungsumgebung

Zur Erstellung von Python-Programmen genügen eigentlich einfache Texteditoren, wie wir sie schon kennengelernt haben, z.B. `vi`, `nano` oder `gedit`. Ein deutlich höheres Komfort-Level und erweiterte Möglichkeiten bieten jedoch so genannte «Integrierte Entwicklungsumgebungen», die dann schon bei der Eingabe des Python-Quelltextes Fehler erkennen oder automatische Vervollständigungen anbieten.

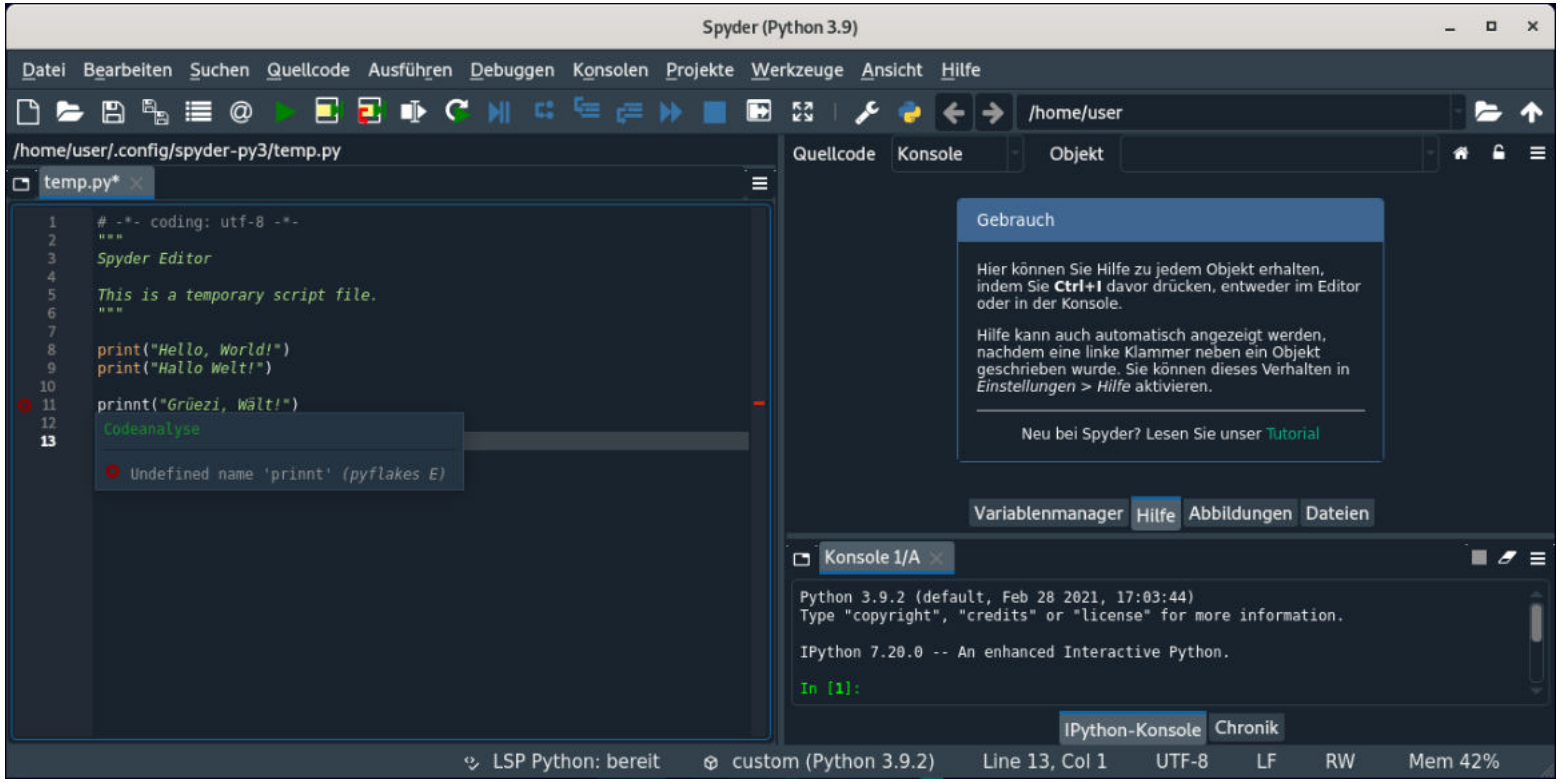
Eine relativ einfache Python-Entwicklungsumgebung ist die «Spyder IDE», siehe https://www.spyder-ide.org.

In der Lernumgebung können Sie einfach das Softwarepaket `spyder3` nach-installieren.

Starten Sie dann das Programm via Desktopumgebung (Dock → Anwendungen anzeigen), ignorieren eventuelle Hinweise zu Updates und stellen die Sprache via «Tools → Preferences → General → Advanced Settings» auf Deutsch um.

Unter Windows und macOS laden Sie das Installationsprogramm von folgender Seite herunter:

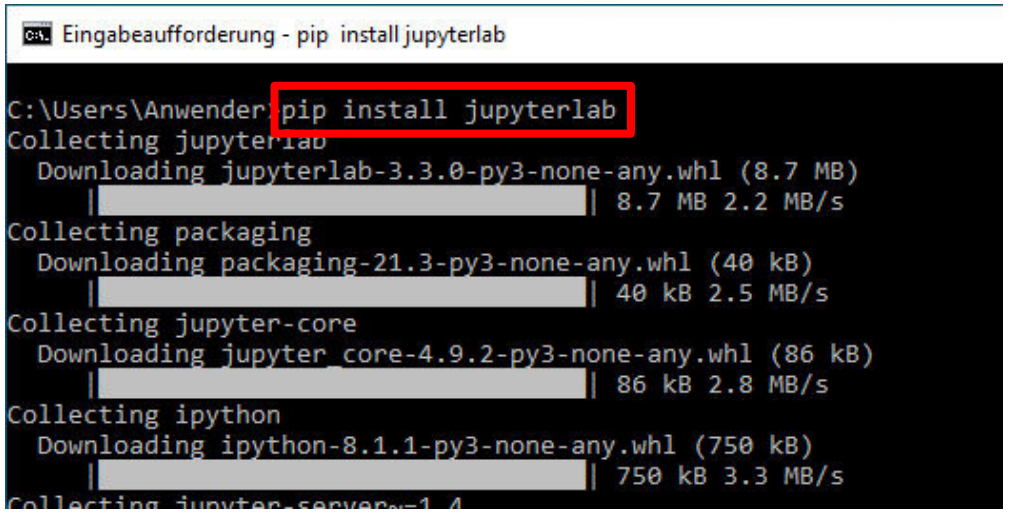
https://github.com/spyder-ide/spyder/releases

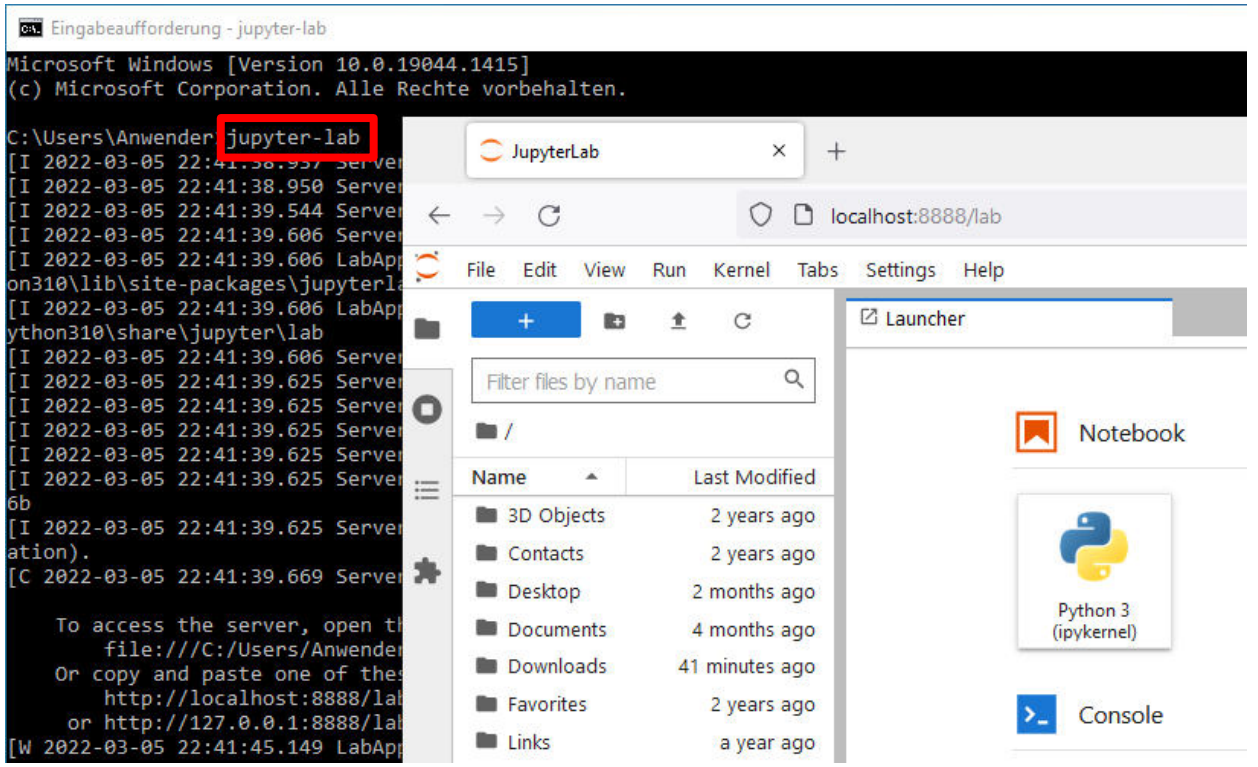


Jupyter-Einführung

JupyterLab via Python-Standardinstallation (Windows)

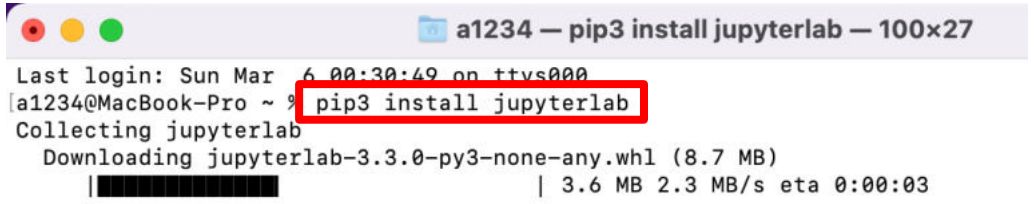
1) Startmenü ▸ Windows-System ▸ Eingabeaufforderung starten und den Befehl `pip install jupyterlab` eingeben.
2) Zum Starten von Jupyterlab kann der Befehl `jupyter-lab` eingegeben werden.

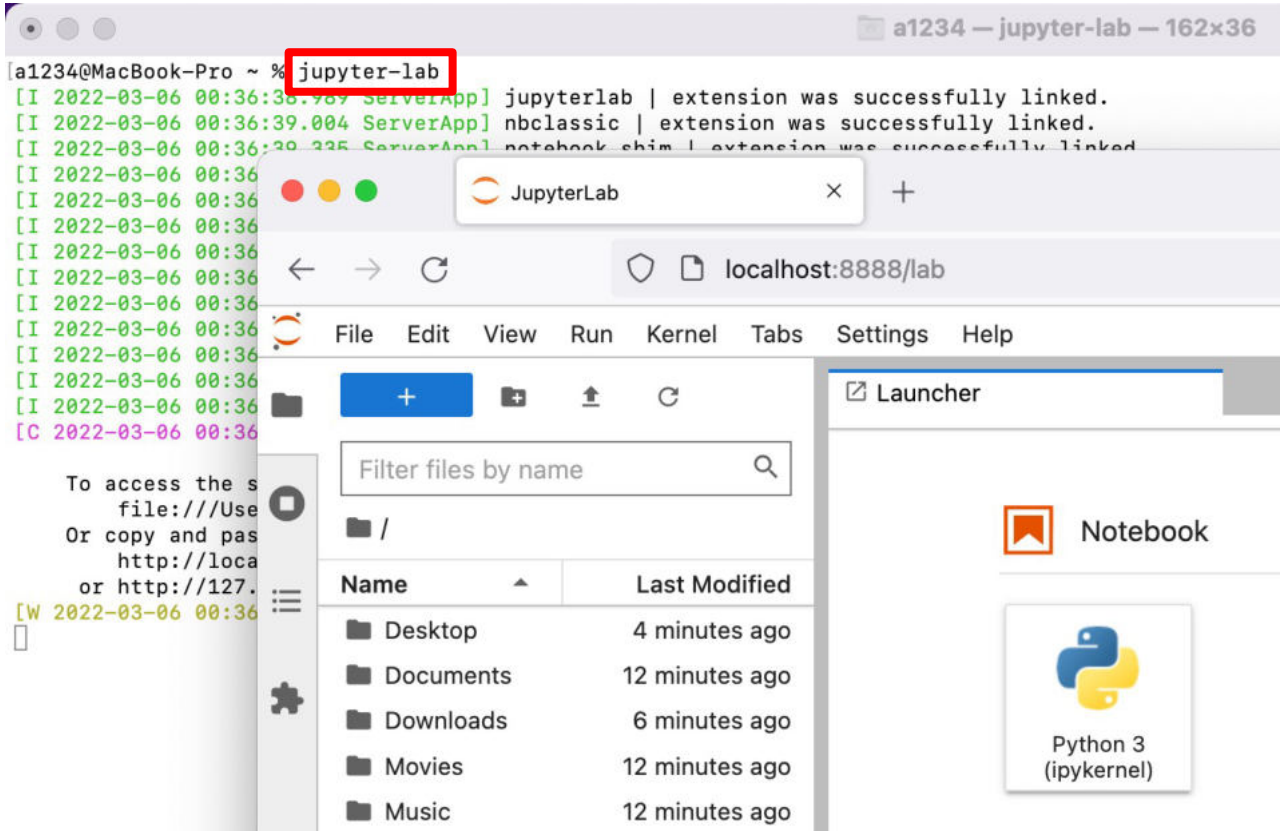




JupyterLab via Python-Standardinstallation (macOS)

1) Im Finder Programme ▸ Dienstprogramme ▸ Terminal starten und den Befehl `pip3 install jupyterlab` eingeben.
2) Zum Starten von Jupyterlab kann der Befehl `jupyter-lab` eingegeben werden.





JupyterLab via Python-Standardinstallation (Linux)

In der Lernumgebung kann JupyterLab via Kommandozeile wie folgt installiert werden:

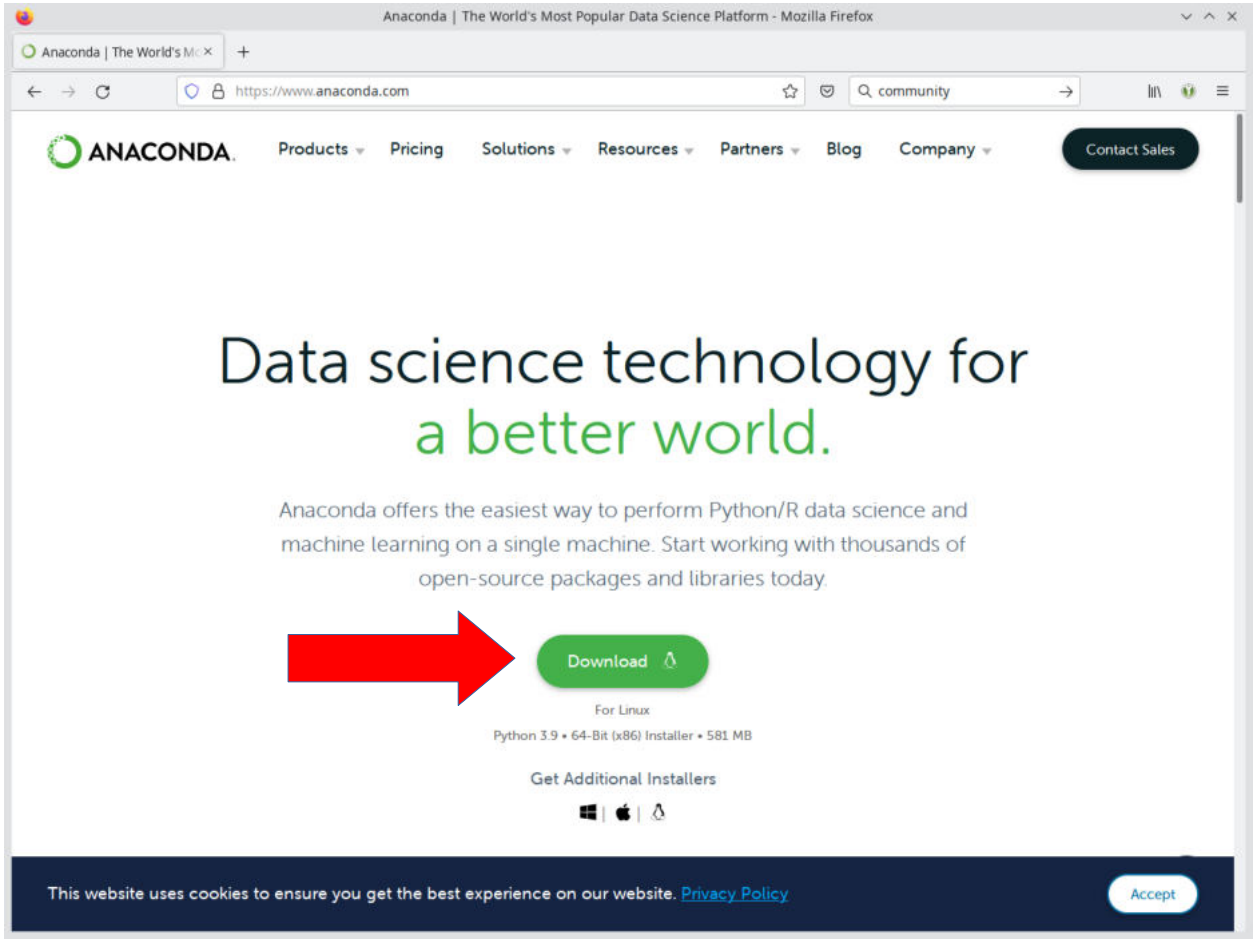
```
sudo apt install python3-pip
```

```
sudo pip3 install jupyterlab
```

Der Start erfolgt dann durch die Eingabe des Befehls:

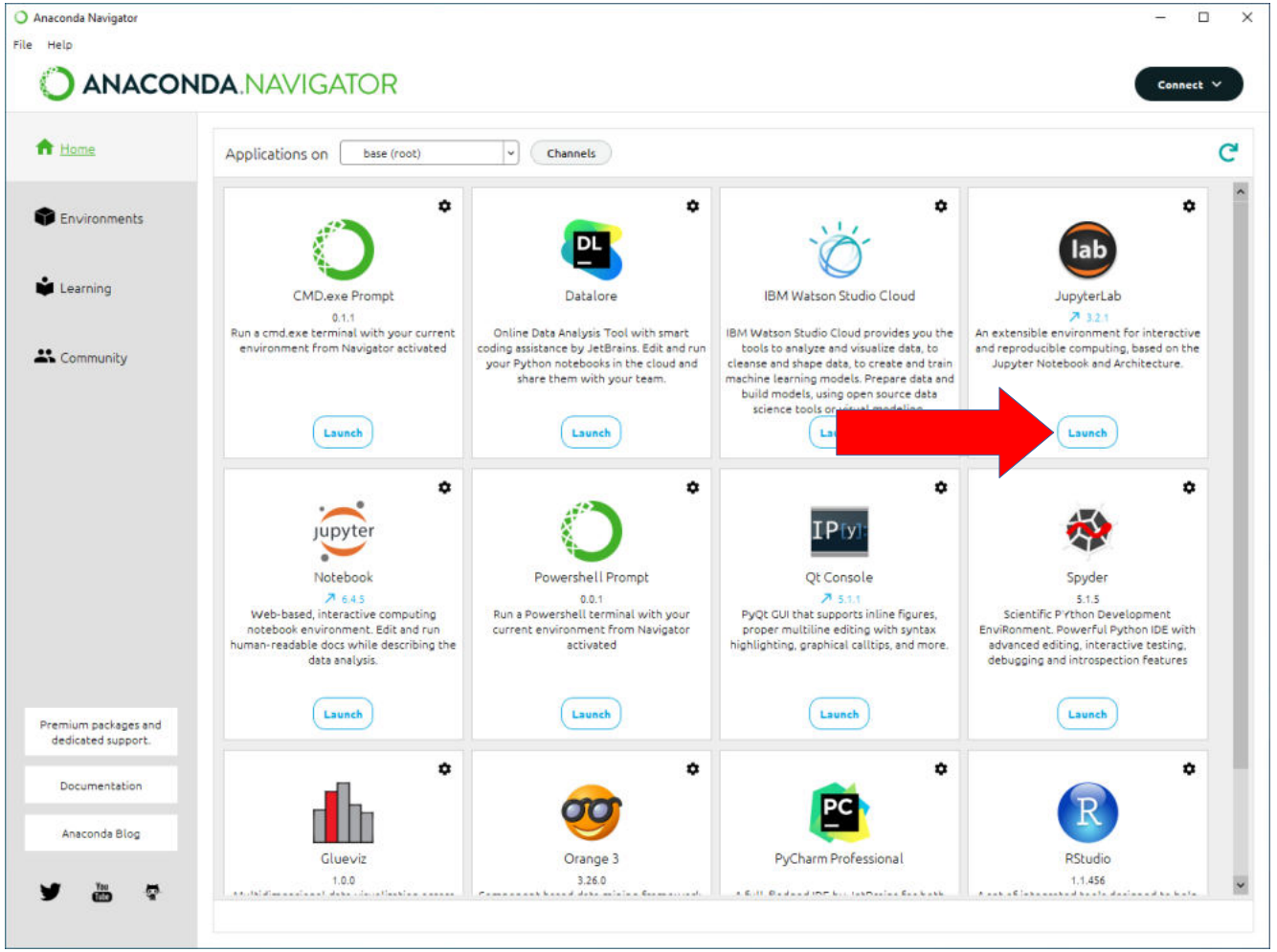
```
jupyter-lab
```

JupyterLab via Anaconda



1) das Installationsprogramm herunterladen
2) das Installationsprogramm starten
3) die Installation und unter Beibehaltung aller Voreinstellungen durchführen

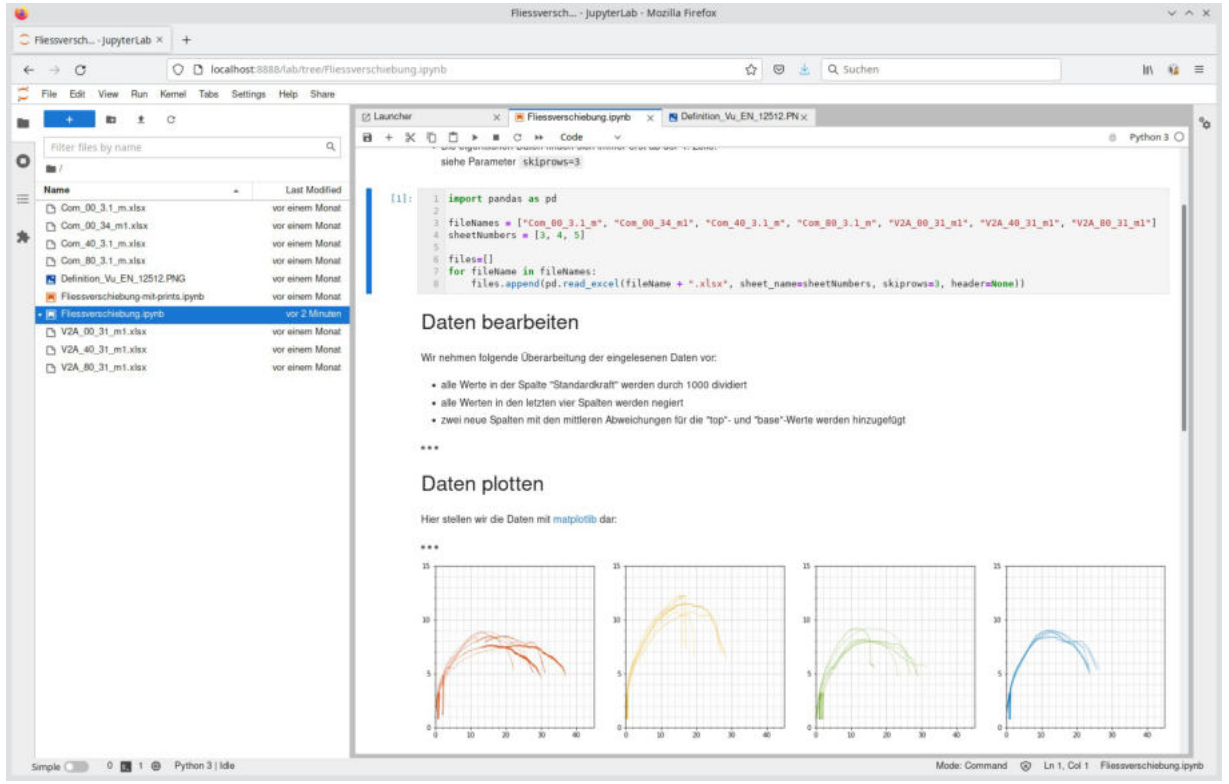
JupyterLab via Anaconda



1) über das Startmenü «Anaconda Navigator» starten

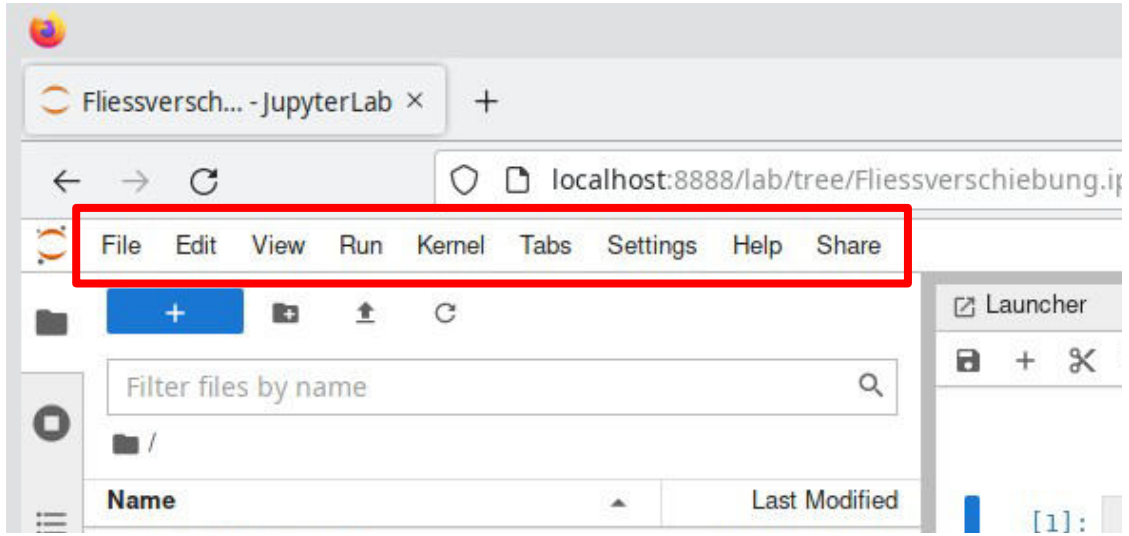
2) JupyterLab starten

JupyterLab



- moderne, flexible und modulare Benutzeroberfläche
- Bearbeitung von Jupyter-Notebooks und vielen weiteren Dateiformaten
- Zusammenstellung und Konfiguration von Data-Science-Workflows

JupyterLab: Menü

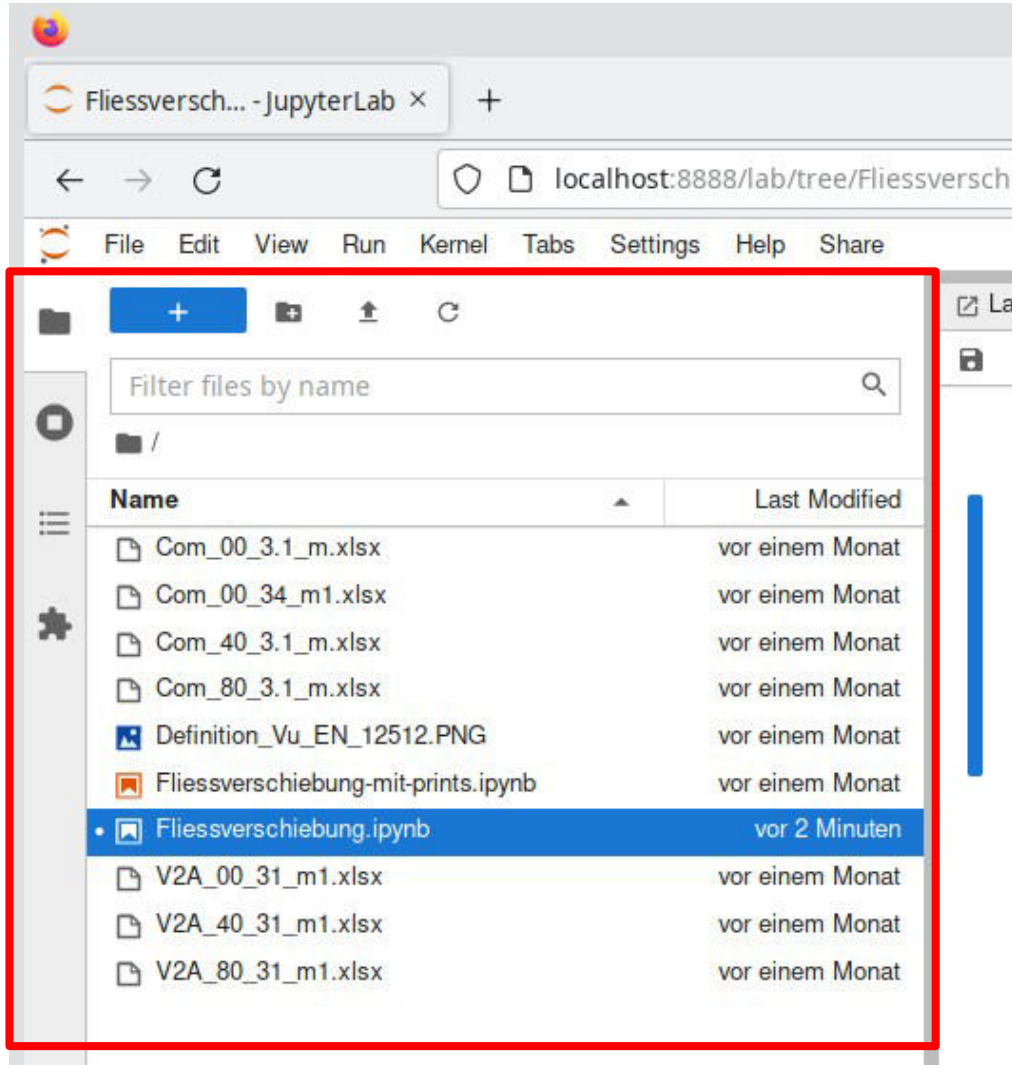


Das **Menü** beinhaltet die wichtigsten Einträge zu:

- Dateioperationen
- Bearbeitung von Dokumenten
- Anpassung des Aussehens von JupyterLab
- Ausführung von verschiedenen Zellen
- Verwaltung von verschiedenen Kerneln
- Wechseln zwischen verschiedenen Dokumenten und Aktivitäten
- Einstellungen
- Hilfe

JupyterLab-Erweiterungen können das Menü um weitere Einträge ergänzen (z. B. im Screenshot der Eintrag «Share» ganz rechts).

JupyterLab: Seitenleiste

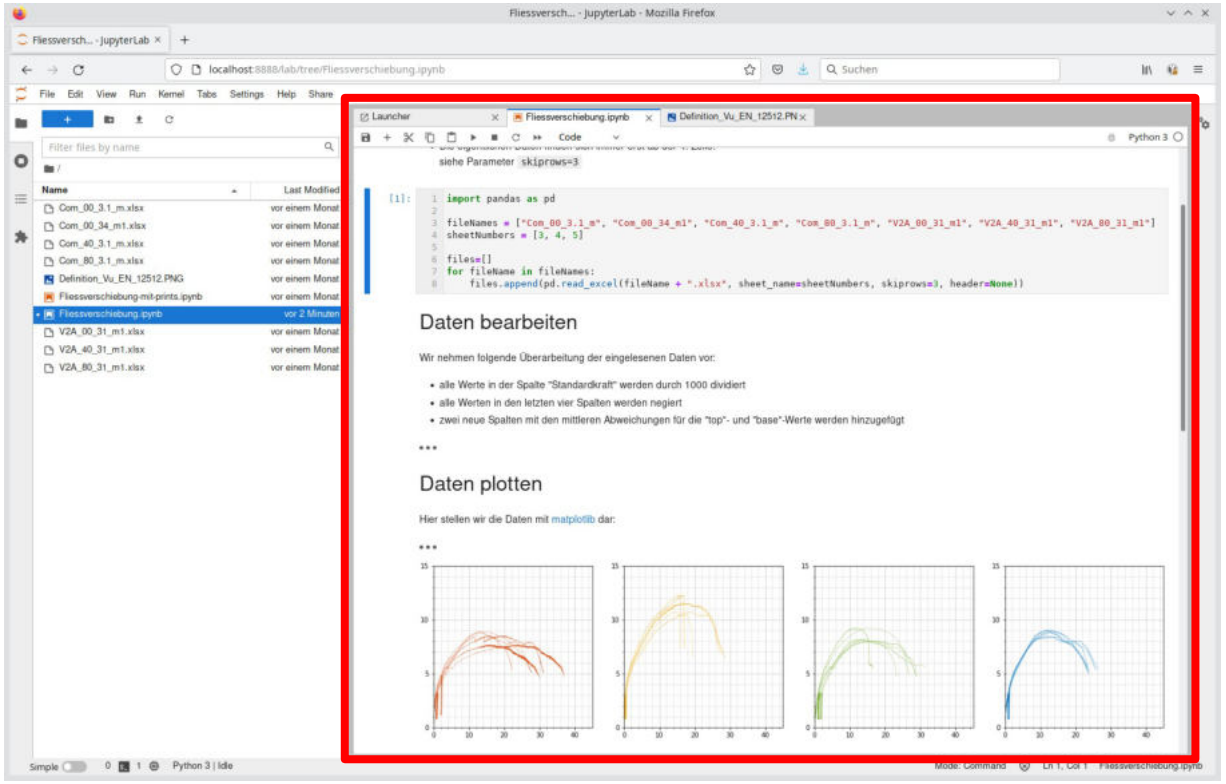


In der **Seitenleiste** befindet sich:

- ein Dateibrowser
- eine Übersicht über alle Dokumente, Kernel und Terminals
- ein Inhaltsverzeichnis zur schnellen Navigation durch ein Jupyter-Notebook
- die Verwaltung der JupyterLab-Erweiterungen

Die Seitenleiste kann durch Anklicken der Icons auf der linken Seite auf- und zugeklappt werden.

JupyterLab: Arbeitsbereich



Im Arbeitsbereich werden

- alle geöffneten Dokumente (Notebooks, Textdateien, Bilder, ...)
- Terminals
- Konsolen

... in sogenannten «Tabs» dargestellt, die sehr flexibel per Drag & Drop oder Kontextmenüs (rechte Maustaste) umsortiert, geteilt und angeordnet werden können.

JupyterLab: Arbeiten innerhalb eines Notebooks

https://jupyterlab.readthedocs.io/en/stable/user/notebook.html

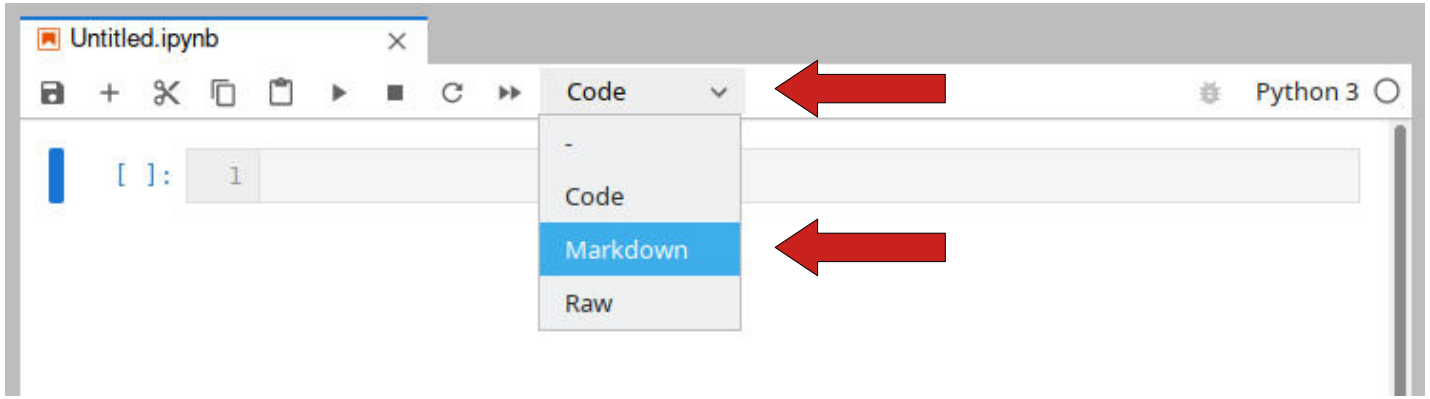
- Neues Notebook anlegen
- Zellen
  - erzeugen
  - navigieren mit Pfeiltasten
  - kopieren (c), ausschneiden (x), einfügen (v)
  - auf- und zuklappen
  - per Drag & Drop umsortieren
  - per Drag & Drop in andere Notebooks kopieren
- synchronisierte Ansichten des gleichen Dokuments erzeugen
- synchronisierte Ansichten für Ausgaben von Zellen erzeugen

Markdown-Zellen

Wenn neue Zellen angelegt werden, sind sie per Voreinstellung «Code-Zellen». Das sind Zellen, in denen Programmcode abgelegt wird, der durch den zugehörigen Kernel (z.B. Python) ausgeführt werden kann.

Für reichhaltig formatierte Kommentare und Erklärungen können in einem Jupyter-Notebook sogenannte **«Markdown-Zellen»** verwendet werden.

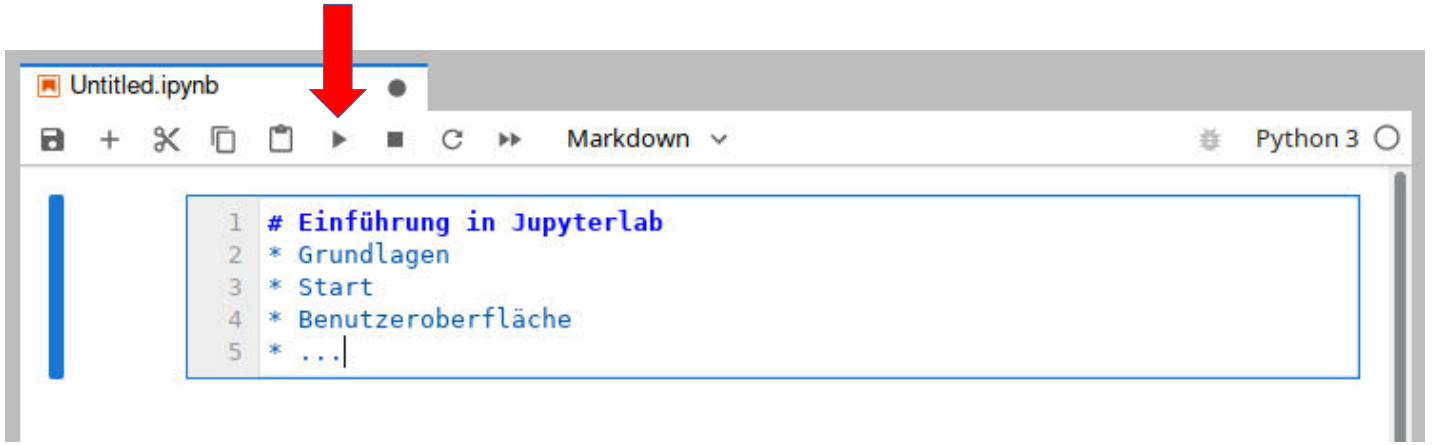
Der Typ einer jeden Zelle kann über das Dropdown-Menü am oberen Rand eines jeden Notebooks umgestellt werden:



(Zellen vom Typ «Raw» werden von Jupyter-Notebooks gar nicht ausgewertet. Sie können für besondere Wünsche beim Exportieren von Notebooks in andere Formate verwendet werden, was wir jedoch hier nicht weiter betrachten werden.)

Markdown-Zellen

Markdown ist eine sehr einfache Auszeichnungssprache, um Texte zu gestalten. Eine gute Übersicht dazu ist im entsprechenden Wikipedia-Artikel zu finden, siehe:
https://de.wikipedia.org/wiki/Markdown



Aufgabe:

Ändern Sie den Typ einer Zelle auf `Markdown` und fügen Sie einen einfachen, jedoch mit Markdown formatierten, Text ein (siehe Wikipedia-Artikel oder Beispiele auf nächster Folie).

Eine Zelle lässt sich unter anderem mit dem Play-Icon (siehe Pfeil ▶ im Screenshot oben), über das Play-Menü oder die Tastenkombination Shift+Enter ausführen.

Wenn Sie die Markdown-Zelle ausführen, wird als Ergebnis der formatierte Text angezeigt.

Kleiner Tipp:

Mit Ctrl+Enter wird die Zelle ausgeführt, ohne dass auf die nächste Zelle gesprungen oder eine neue Zelle angelegt wird.

Markdown-Beispiele

```
# Überschrift 1
## Überschrift 2
### Überschrift 3, usw.

**Fett**

*Kursiv*

Zitat:
> Wer Wind säht, wird Sturm ernten.

horizontale Linie:
- - -

Das hier ist `Inline-Quelltext`, der innerhalb einer Zeile eingefügt wird.

Hier ist ein ganzer Code-Block:
```python
message="Hello World!"
print(message)
```
- das hier ist eine
- unnummerierte Liste

1. und hier
2. ist noch eine
3. nummerierte Liste

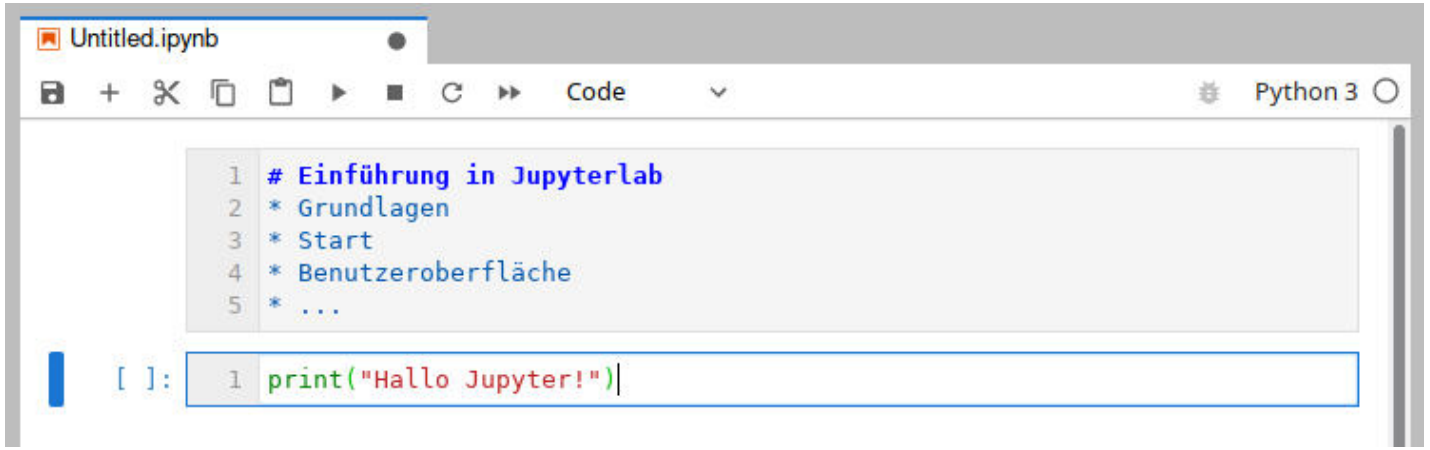
ein Hyperlink zur [BFH](https://www.bfh.ch)

und der Link zu einem Bild:
![BFH-Logo](https://upload.wikimedia.org/wikipedia/commons/a/a2/BFH_Logo_deutsch.png)
```

Links sehen Sie ein paar typische Beispiele für mit Markdown formatierte Texte.

Das Ergebnis wird erst nach der Ausführung der Zelle (z.B. mit Shift+Enter) angezeigt.

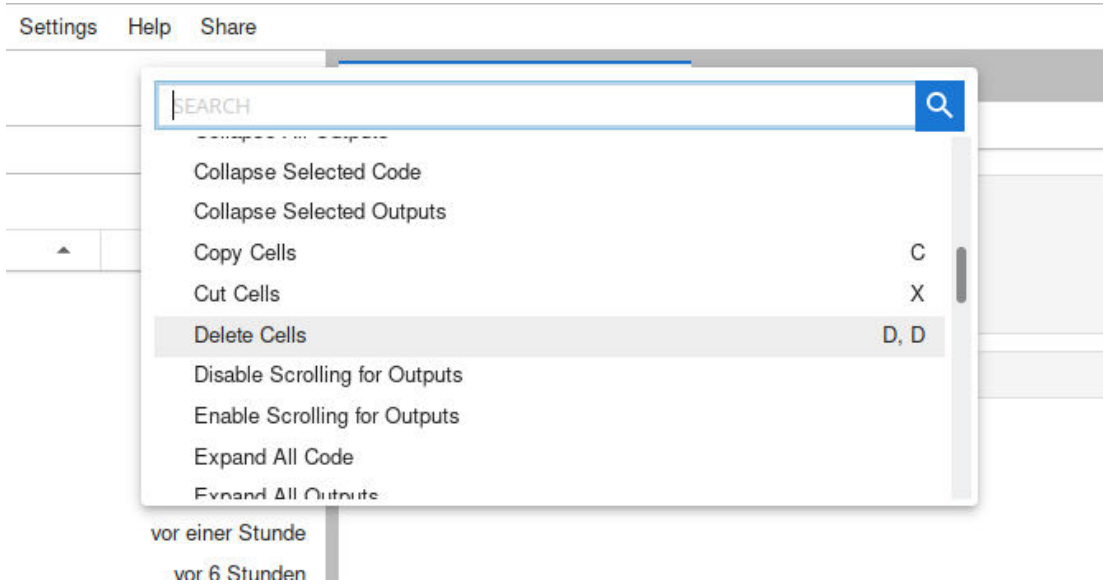
Einschub: Edit- und Command-Mode



Vielleicht ist es Ihnen schon aufgefallen: Die Zelle, die Sie gerade bearbeiten, bekommt einen kleinen blauen Rahmen. Dann befindet sich die Zelle im so genannten **Edit-Mode**, wobei alle Tastatureingaben dem Inhalt der Zelle hinzugefügt werden.

Der Edit-Mode kann durch Drücken der ESC-Taste beendet werden. Dadurch wird in den **Command-Mode** gewechselt, wobei dann alle Tastatureingaben als *JupyterLab-Befehle* interpretiert werden.

Einschub: Edit- und Command-Mode



Welche Befehle und entsprechende Tastaturkürzel zur Verfügung stehen, können Sie sich durch den Aufruf von

`View ▸ Activate Command Palette`

anzeigen lassen.

Aufgaben:

Wechseln Sie in den Command-Mode und fügen Sie unter der aktuellen Zelle eine neue Zelle über das entsprechende Tastaturkürzel hinzu (siehe Command Palette!).

Wechseln Sie mit dem entsprechenden Tastaturkürzel in den Edit-Mode und fügen Sie der neuen Zelle einen kurzen, beliebigen Inhalt hinzu.

Löschen Sie die neue Zelle über das entsprechende Tastaturkürzel.

Mathematische Formeln

Mathematische Formeln können in Markdown-Zellen mitten im Text zwischen zwei Dollar-Zeichen als LaTeX-Formeln eingegeben werden, siehe z.B.: https://en.wikibooks.org/wiki/LaTeX/Mathematics

Soll eine Formel auf einer Zeile allein dargestellt werden, müssen doppelte Dollar-Zeichen verwendet werden:

```
1 Die Formel $M_{max}=q \cdot l^2/8$ bedeutet, dass das maximale Biegemoment eines
  Einfeldträgers mit Gleichlast in Feldmitte  $M_{max}$ dadurch ermittelt werden kann,
  indem das Produkt der Streckenlast $q$ mit dem Quadrat der Stützweite des Trägers $l$
  durch 8 dividiert wird.
2
3 Die Formel kann selbstverständlich auch schöner und allein auf einer Zeile dargestellt
  werden: $$M_{max}=\frac{q \cdot l^2}{8}$$
```

Aufgabe:

Geben Sie eine andere Ihnen bekannte Formel in einer Markdown-Zelle ein und lassen Sie sie durch «Ausführen» der Zelle darstellen.

Die Formel $M_{max} = q \cdot l^2/8$ bedeutet, dass das maximale Biegemoment eines Einfeldträgers mit Gleichlast in Feldmitte M_{max} dadurch ermittelt werden kann, indem das Produkt der Streckenlast q mit dem Quadrat der Stützweite des Trägers l durch 8 dividiert wird.

Die Formel kann selbstverständlich auch schöner und allein auf einer Zeile dargestellt werden:

$$M_{max} = \frac{q \cdot l^2}{8}$$

Codezellen:

Führen Sie ein paar einfache Python-Befehle einzeln oder gemeinsam in verschiedenen Jupyter-Codezellen aus.

Anregungen:

```
print("Hello, World!")

print(42)

a = 13
a += 2
print(a)
```

Zellennummerierung

Vor den Codezellen erscheinen sich ständig ändernde Nummern. Diese zeigen die Ausführungsreihenfolge der jeweilige Codezellen an:

wurde als 30. Codezelle ausgeführt

```
[30]:
1 a = 13
2 a += 2
3 print(a)
```

```
15
```

wurde als 29. Codezelle ausgeführt (also ***vor*** der oberen Zelle!)

```
[29]:
1 fruits = ["Äpfel", "Bananen", "Kirschen"]
2 for x in fruits:
3     print("Hmm, lecker " + x  + "!")
4     print("Gibt's noch mehr?")
5 print("Ach schade, nichts mehr da.")
```

```
Hmm, lecker Äpfel!
Gibt's noch mehr?
Hmm, lecker Bananen!
Gibt's noch mehr?
Hmm, lecker Kirschen!
Gibt's noch mehr?
Ach schade, nichts mehr da.
```

wird **soeben** ausgeführt

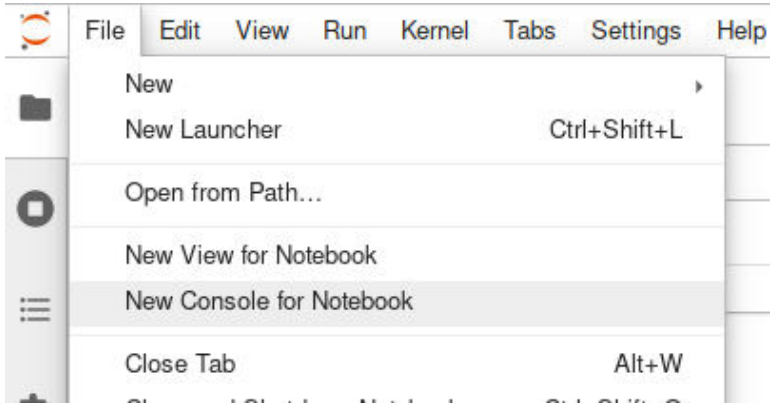
```
[*]:
1 while True:
2     pass
```

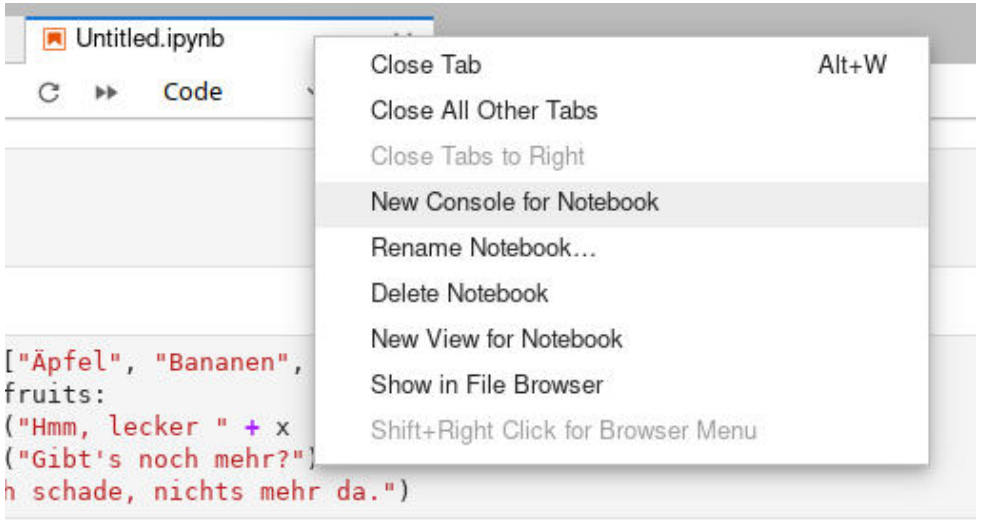
wurde **noch nie** ausgeführt

```
[ ]:
1 print(42)
```

Einschub: Console

Die Console bietet die Möglichkeit, ausserhalb des eigentlichen Notebooks schnell und einfach Befehle im gleichen Kernel laufen zu lassen. Die Console kann unter anderem über das Menü «File ▸ New Console for Notebook» oder über das Kontextmenü (rechte Maustaste) des Notebook-Tabs gestartet werden:

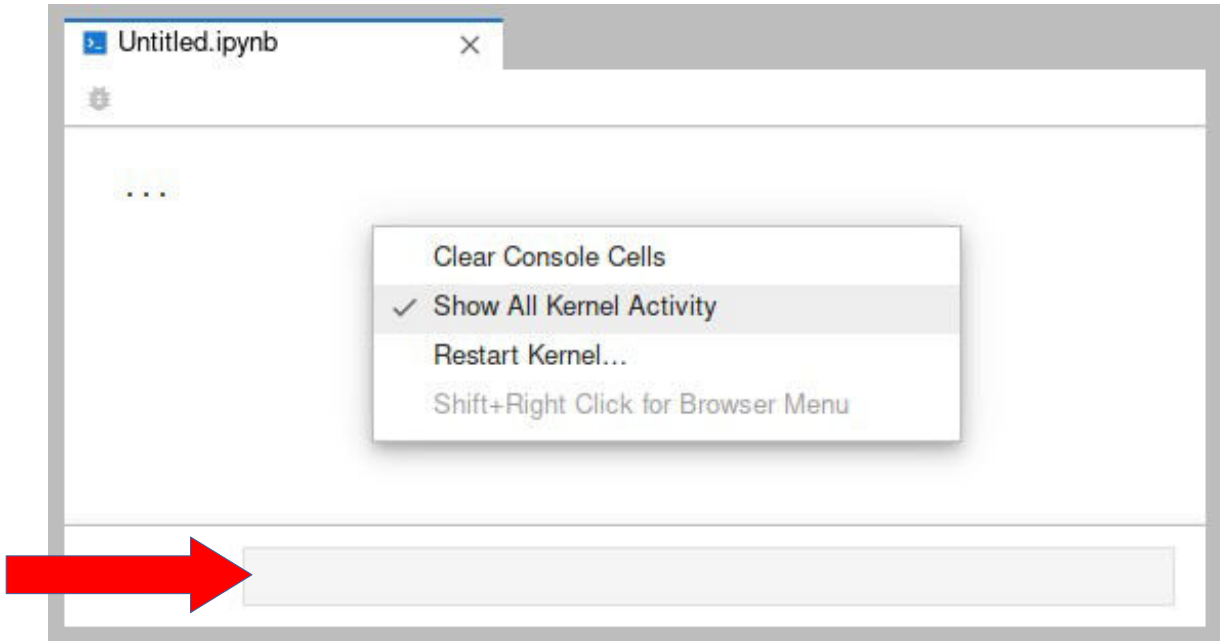




Einschub: Console

Zusätzlich bietet die Console noch die Möglichkeit, alle im Kernel ausgeführten Befehle und deren Ausgeben in ihrer tatsächlichen Reihenfolge zu sehen. Die Aktivierung erfolgt via Kontextmenü ▸ «Show All Kernel Activity».

Codezelle für Experimente ausserhalb des Notebooks (Ausführung mit `Shift+Enter`)



Aufgabe: Console

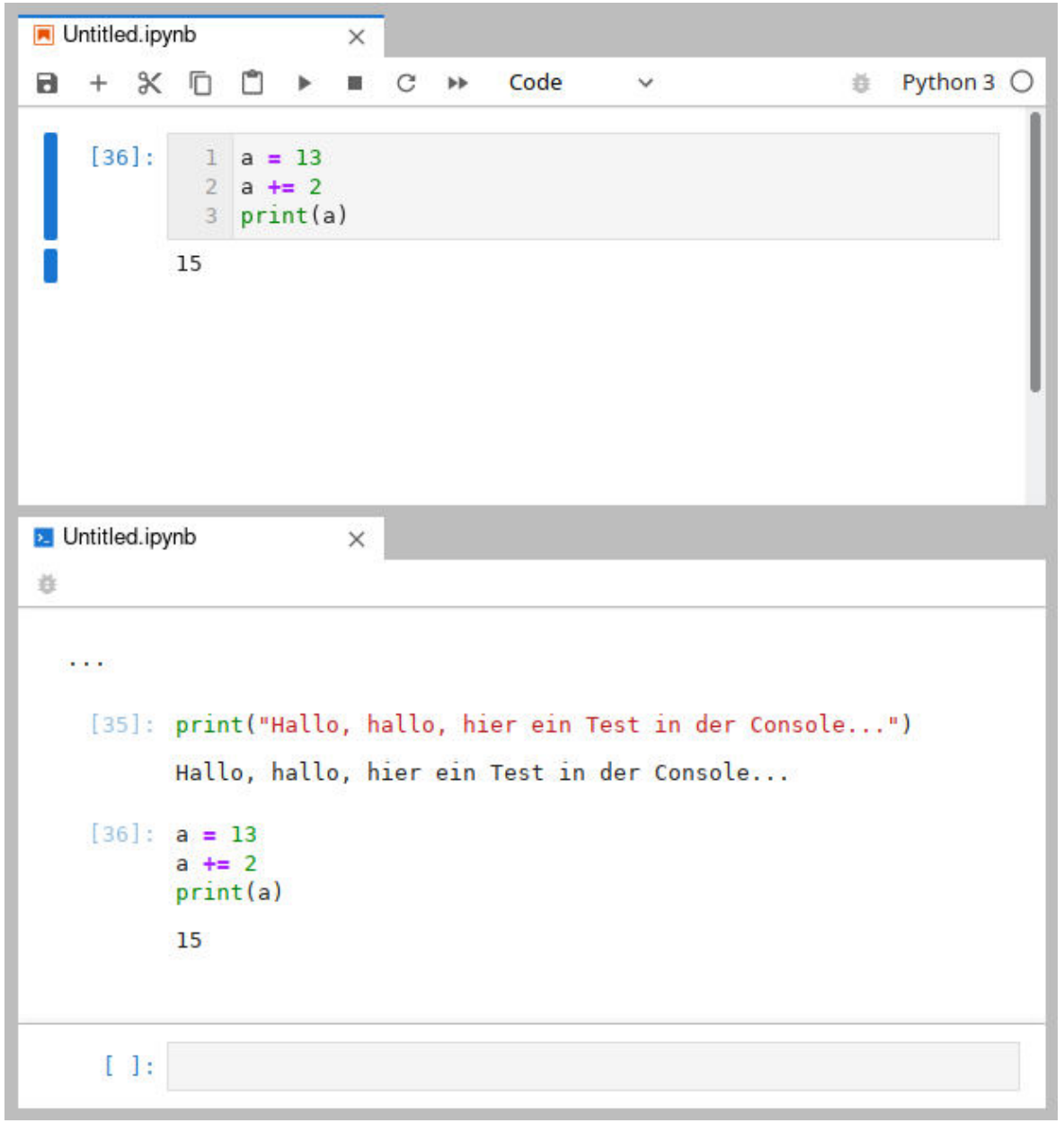
Aktivieren Sie die Console und die Ausgabe sämtlicher Kernelaktivitäten (siehe vorherige Folie).

Führen Sie sowohl die Codezelle der Console als auch Codezellen Ihres Notebooks aus und beobachten Sie sowohl die Zellennummerierung im Notebook aus auch die Ausgaben in der Console.

Tipp:

Die Codezelle am unteren Rad der Console hat eine History-Funktion, d.h. wenn die Codezelle leer ist, kann man mit den Pfeiltasten (hoch und runter) durch die bisher ausgeführten Befehle navigieren und diese über diesen Weg auch einfach anpassen und erneut ausführen.

Bitte ausprobieren!



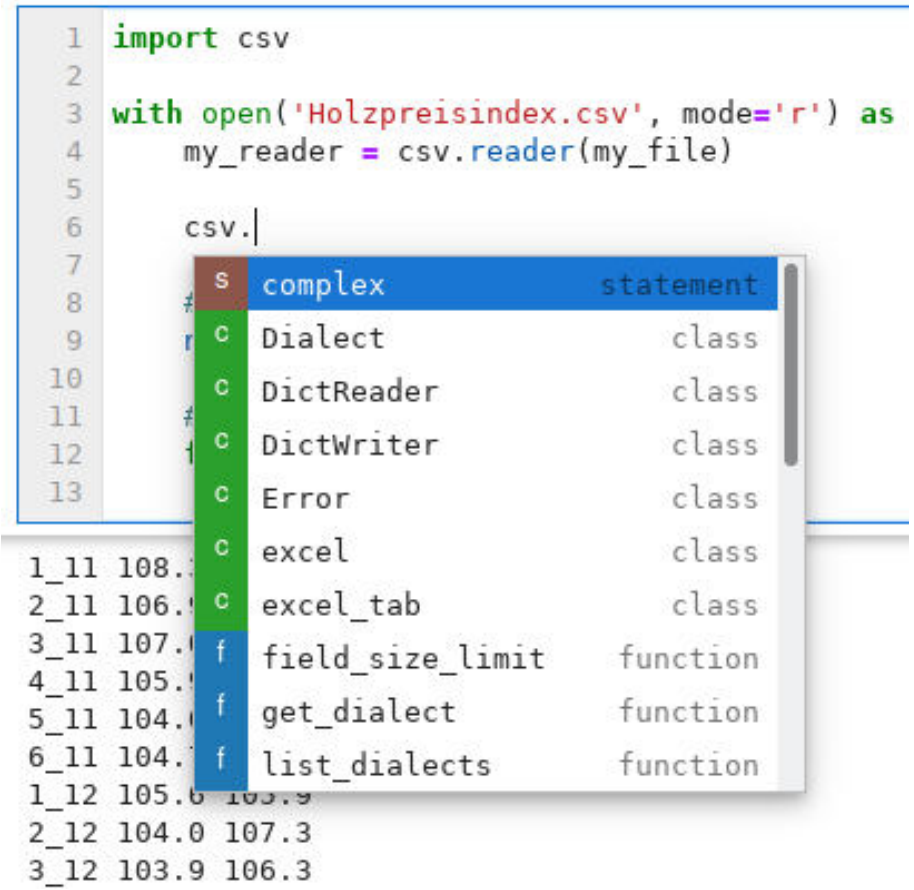
Tipps & Tricks

Durch Drücken der **TAB-Taste** kann **Code automatisch vervollständigt** werden.

Rechts ein Beispiel, bei dem direkt nach der Eingabe des Textes «csv.» die TAB-Taste gedrückt wurde. Dann werden alle Methoden aus dem csv-Modul angezeigt.

Falls der Code an der Stelle schon eindeutig war, z.B. wenn Sie «csv.rea» eingeben und dann die TAB-Taste drücken, wird automatisch auf «csv.reader» vervollständigt, da es die einzige Methode ist, die mit diesem Präfix anfängt.

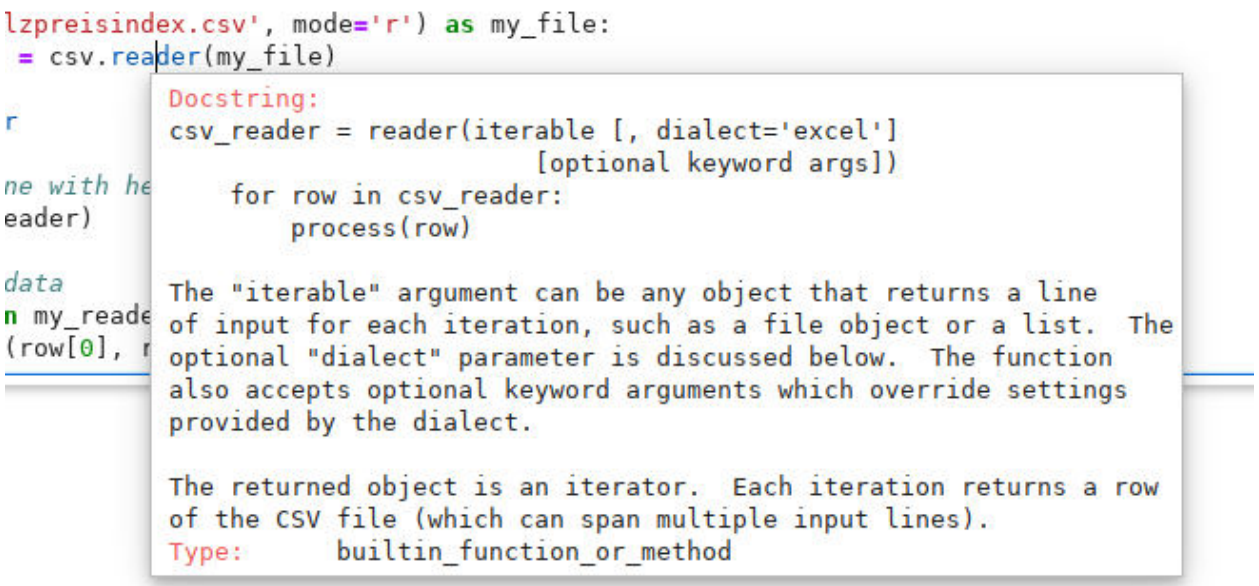
Bitte ausprobieren!



Tipps & Tricks: Dokumentation

Die Python-Module und ihre Funktionen sind im Allgemeinen sehr gut auf ihren jeweiligen Webseiten dokumentiert. Hier nochmal das Beispiel für das Modul csv:
https://docs.python.org/3/library/csv.html

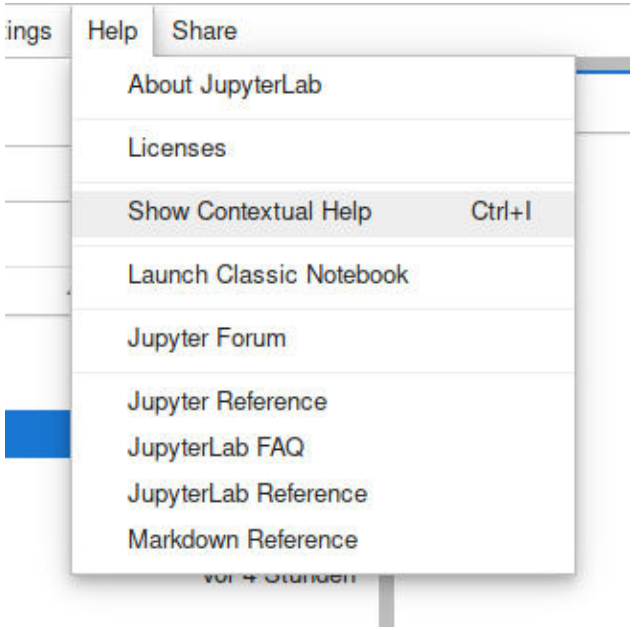
Um aber nicht ständig zwischen Jupyter und der Dokumentation im Webbrowser wechseln zu müssen, kann man sich die **Dokumentation** eines Moduls oder einer Funktion, auf der der Textcursor gerade steht, schnell und direkt mit der Tastenkombination **Shift-TAB** anzeigen lassen (rechts das Beispiel für die Funktion reader im Modul csv).



Aufgabe:

Lassen Sie sich auf diese Art und Weise die Dokumentation des Moduls csv selbst anzeigen.

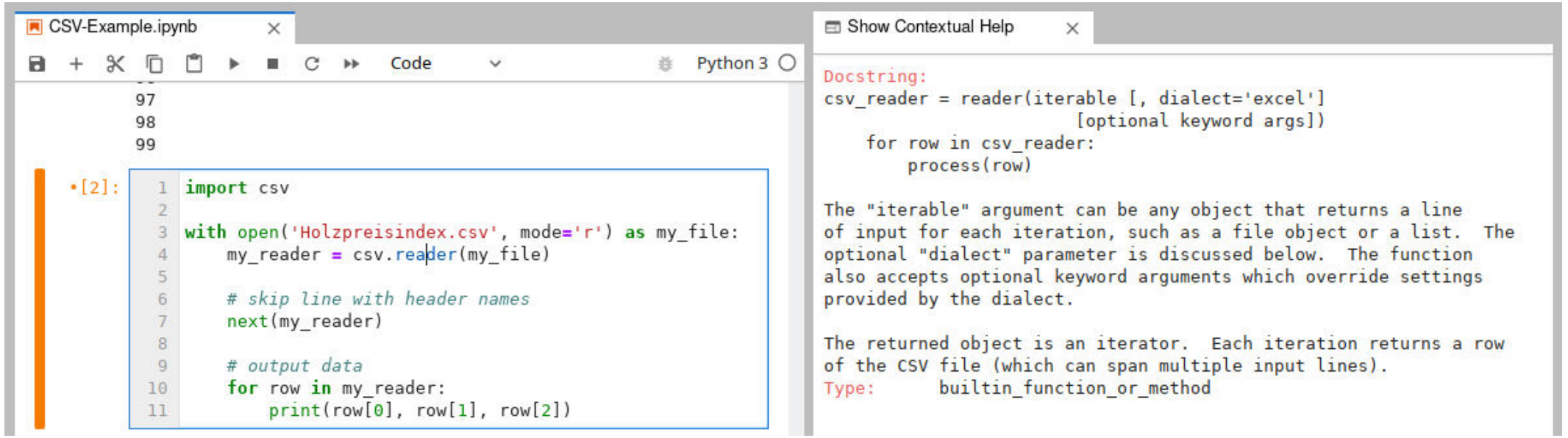
Tipps & Tricks: Kontexthilfe



Über das Menü «Help ▸ Show Contextual Help» oder mit der Tastenkombination **`Ctrl+I`** wird die Kontexthilfe eingeblendet. In diesem Hilfefenster wird dynamisch immer die Dokumentation des Moduls oder der Funktion angezeigt, in der sich der Textcursor gerade befindet.

(Das Modul oder die Funktion muss jedoch schon mindestens einmal geladen worden sein, die Zelle also sicherheitshalber nochmal ausführen.)

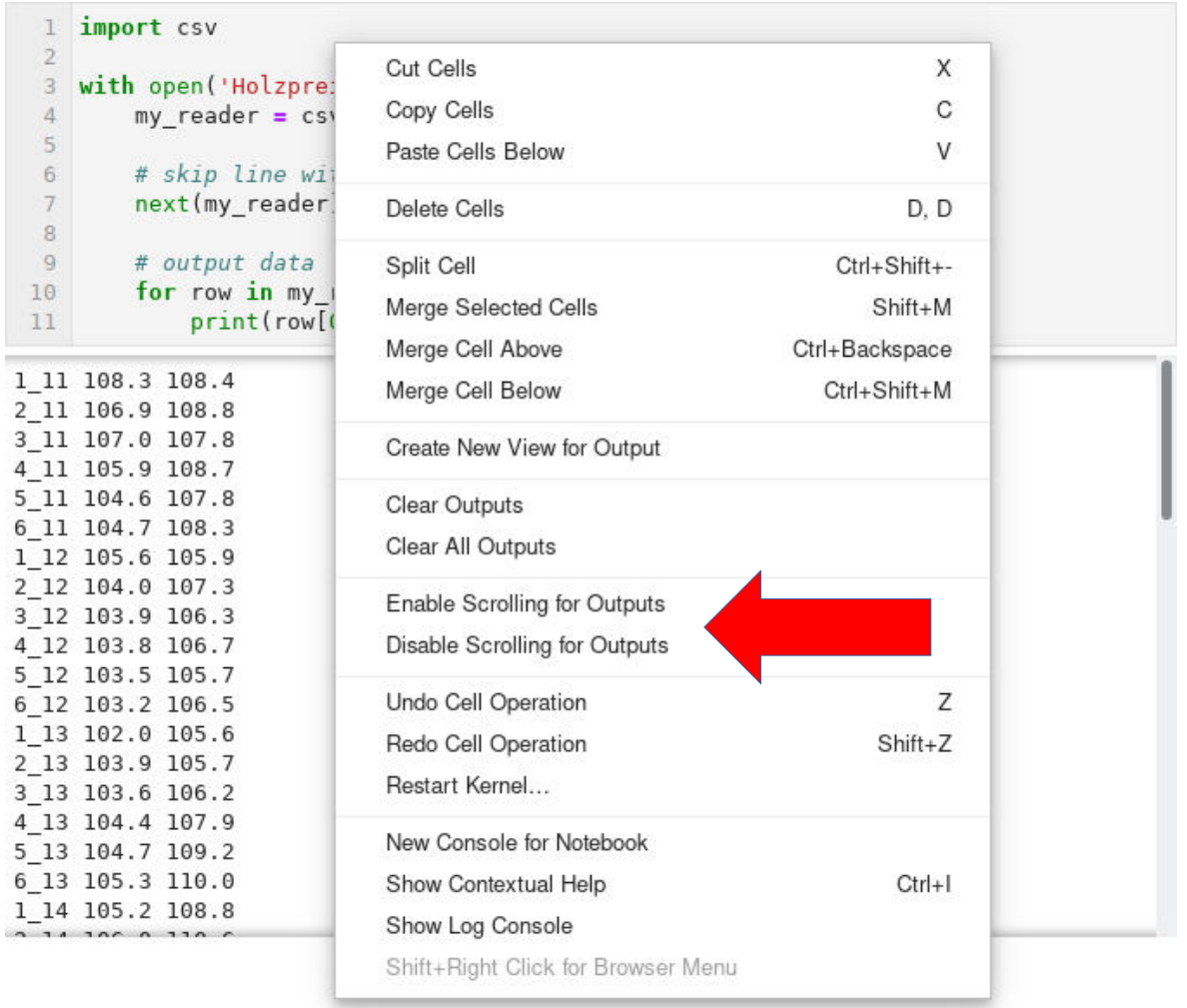
Bitte ausprobieren!



Tipps & Tricks: Scrolling von langen Ausgaben

Es kann vorkommen, dass die Ausgabe einer Codezelle sehr lang wird. Damit die Übersicht und die Navigation im Notebook nicht darunter leidet, kann für jede Zelle festgelegt werden, ob die Ausgaben vollständig oder nur in einem kleinen scrollbaren Abschnitt angezeigt werden.

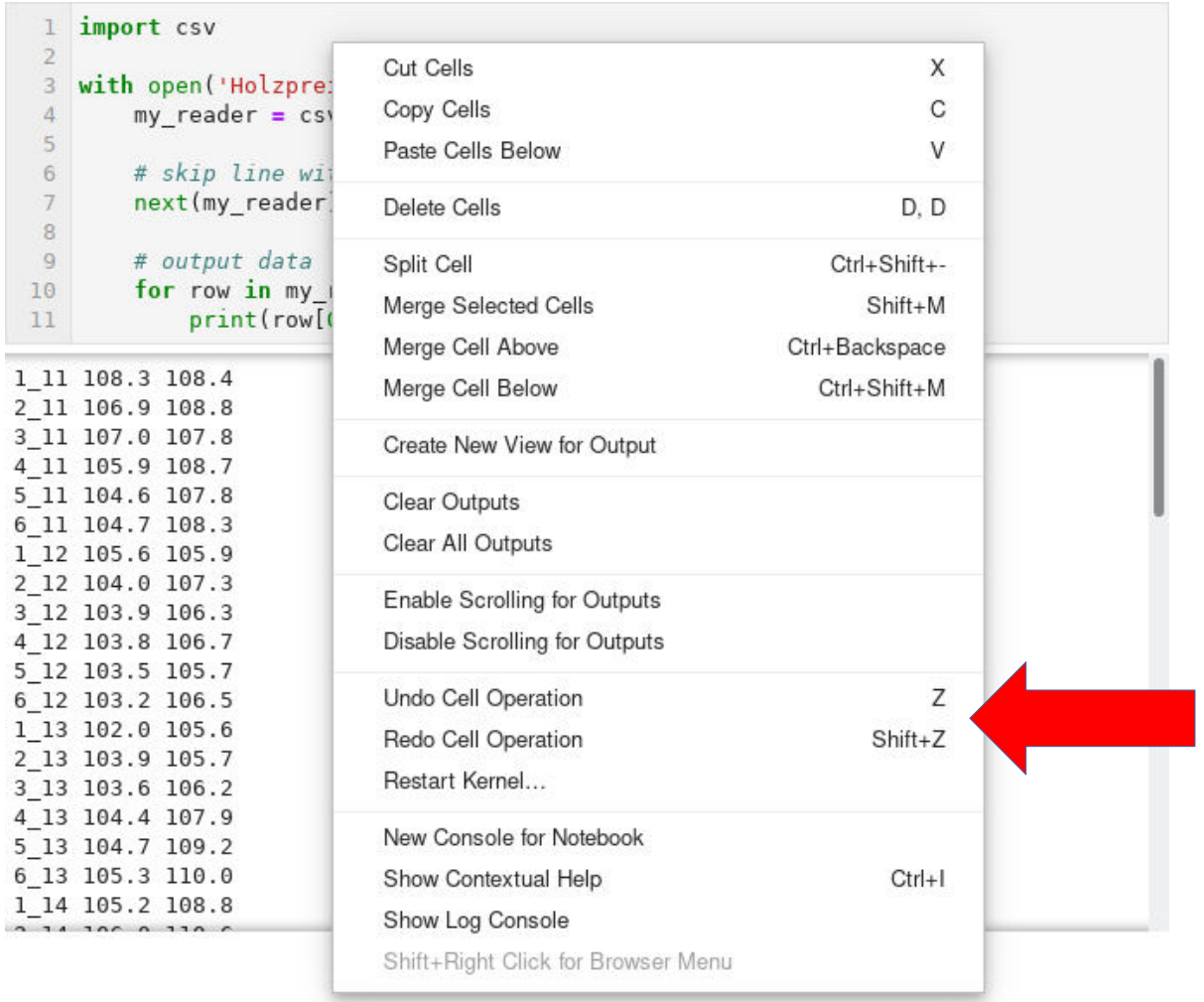
Diese Funktion erreicht man über das Kontextmenü (rechte Maustaste) einer jeden Zelle.



Tipps & Tricks: Rückgängig / Wiederherstellen

Während man eine Zelle im Editiermodus bearbeitet, funktionieren die Operationen *Rückgängig* und *Wiederherstellen* wie in jeder üblichen Textverarbeitung über die Tastaturkombinationen **`Ctrl+Z`** und **`Ctrl+Y`**.

Im Commandmodus geht es etwas schneller mit nur **`Z`** und **`Shift+Z`**.

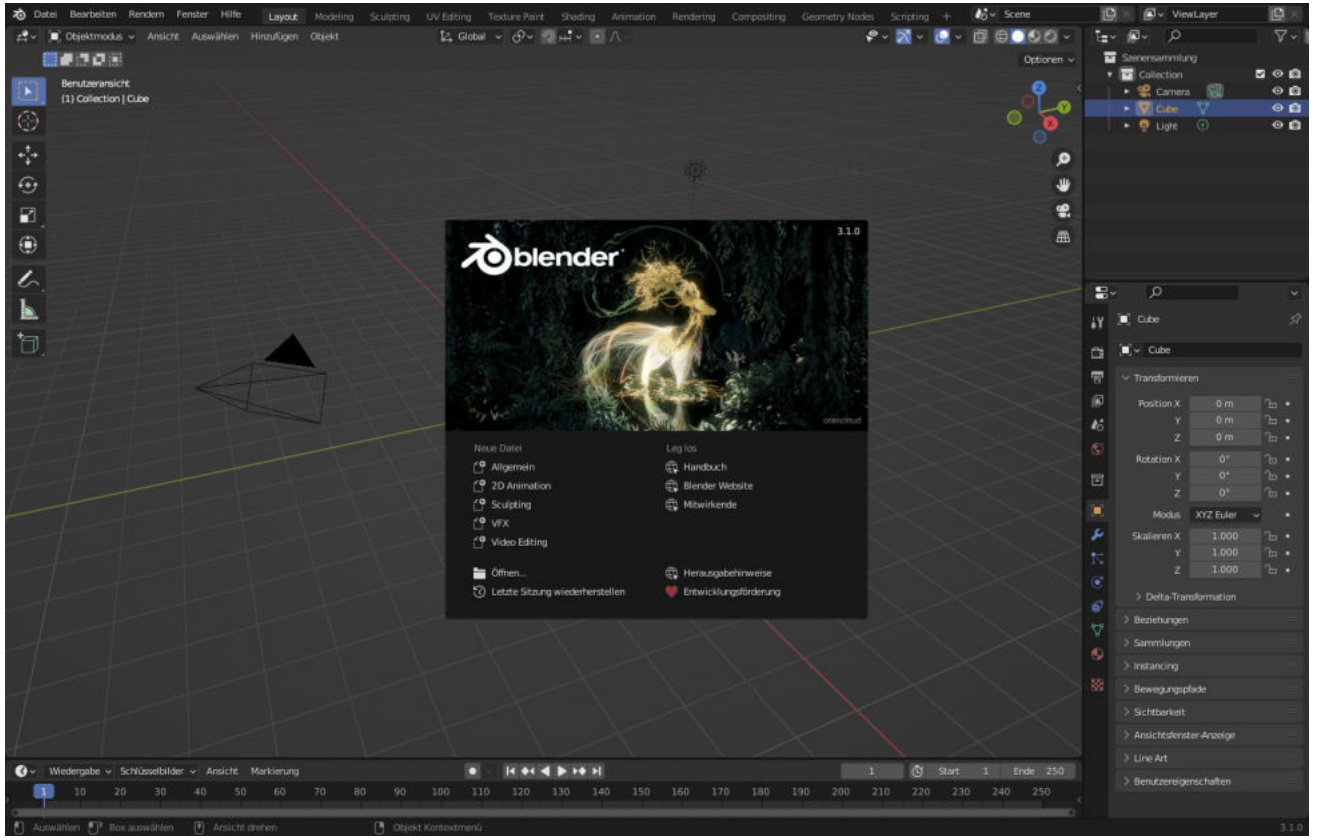


Blender

Einführung Blender

Zur Erzeugung hochqualitativer grafischer Darstellungen von CAD-Zeichnungen werden darauf spezialisierte Programme, sogenannte 3D-Grafiksuiten, verwendet.

Blender ist eine freie und leistungsstarke Software zur Erstellung von dreidimensionalen Grafiken, Animationen und Spielen, siehe:
https://www.blender.org



FreeCAD
(optional)

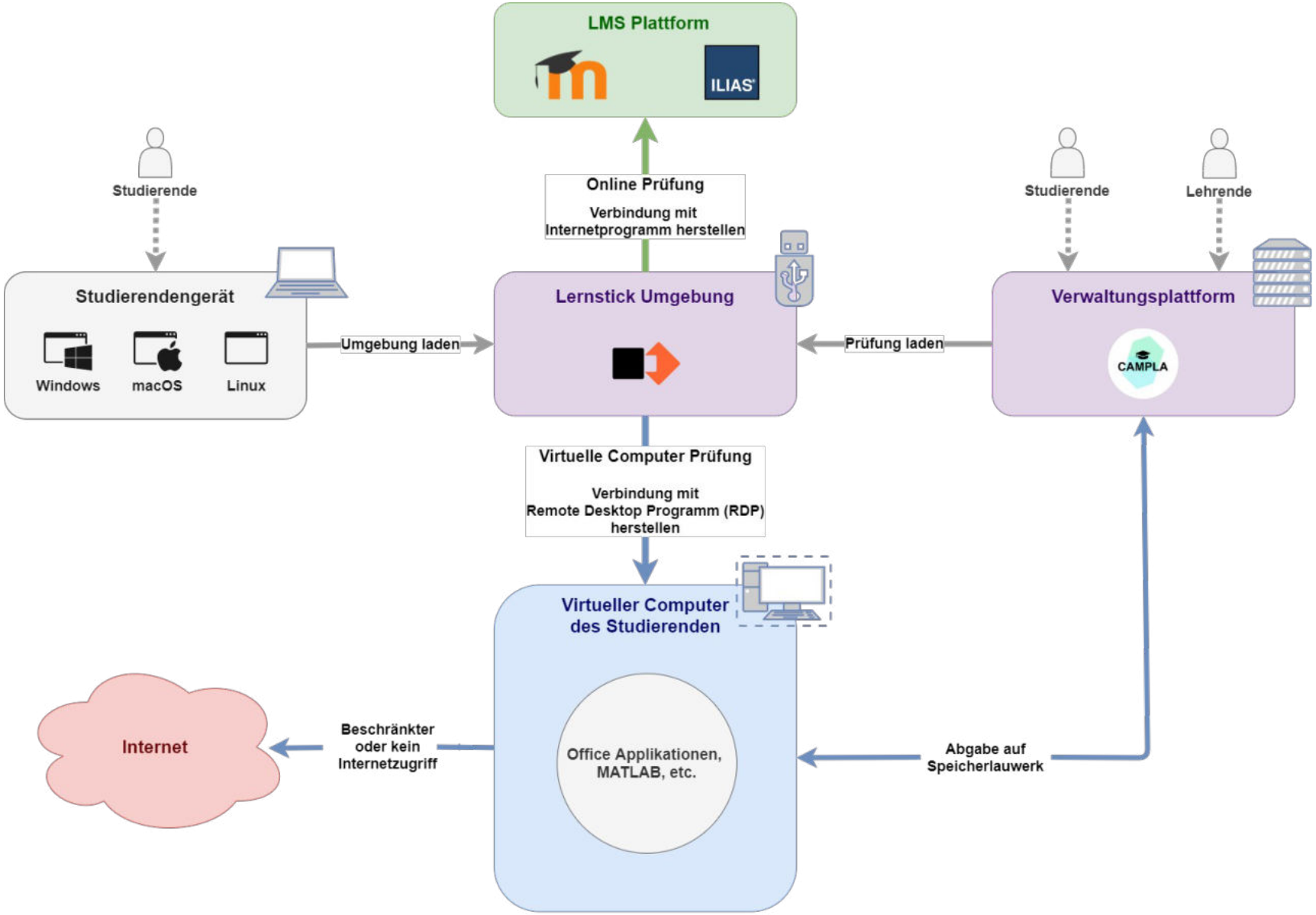
Einfaches CAD-Programm? → FreeCAD

https://www.freecad.org

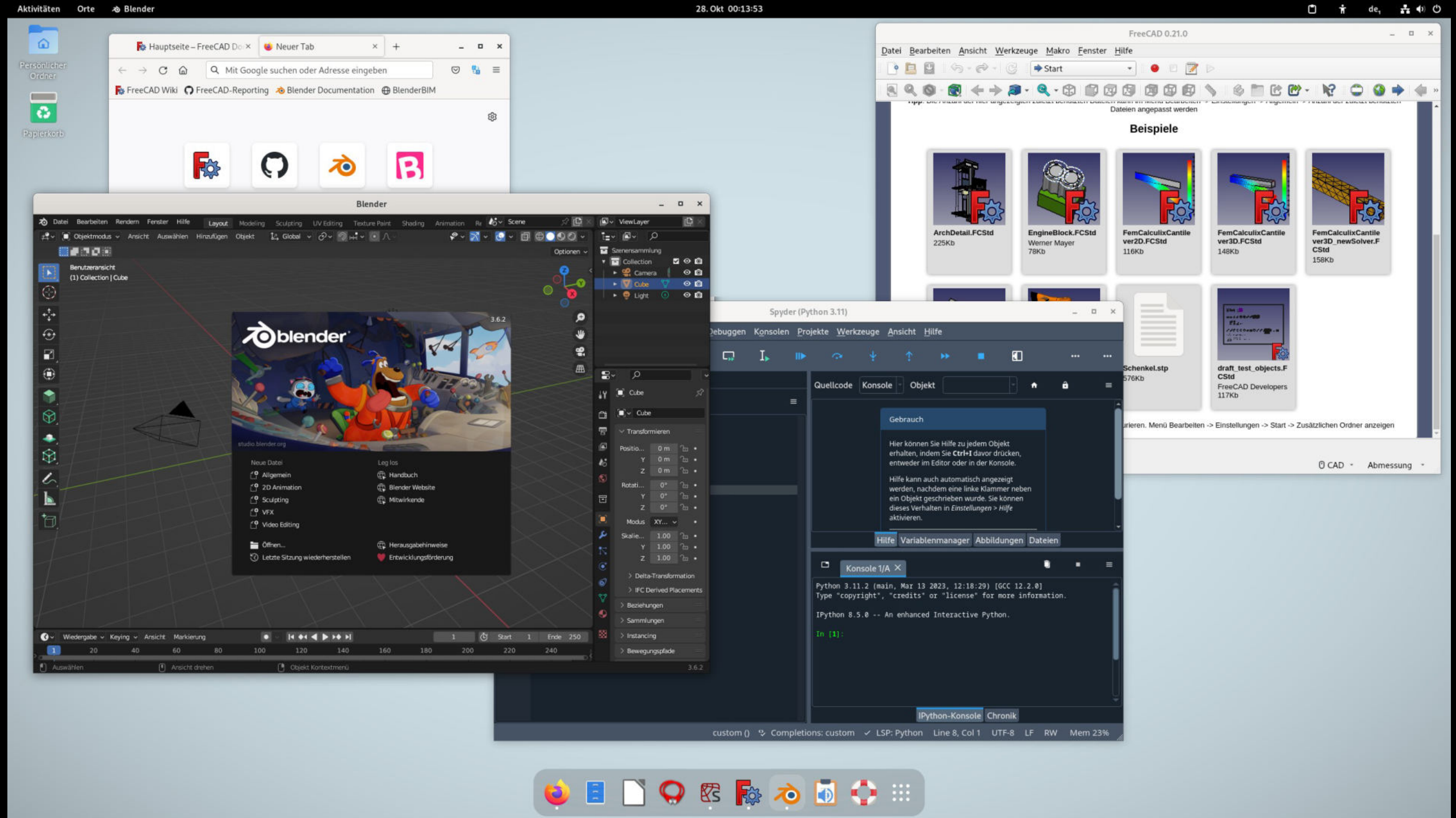
CAMPLA / Lernstick

CAMPLA / Lernstick

- Verwaltungsoberfläche CAMPLA
  - Zugriff auf die Prüfung direkt geht leider nur im BFH-Netzwerk…
- Lernstick ohne CAMPLA



CAMPLA Systemlandschaft


Aktivitäten
Orte
Blender
28. Okt 00:13:53
de
Persönlicher Ordner
Papierkorb
Hauptseite – FreeCAD Do
Neuer Tab
Mit Google suchen oder Adresse eingeben
FreeCAD Wiki
FreeCAD-Reporting
Blender Documentation
BlenderBIM
FreeCAD 0.21.0
Datei Bearbeiten Ansicht Werkzeuge Makro Fenster Hilfe
Start
Dateien angepasst werden
Beispiele
ArchDetail.FCStd
225Kb
EngineBlock.FCStd
Werner Mayer
78Kb
FemCalculixCantilever2D.FCStd
116Kb
FemCalculixCantilever3D.FCStd
148Kb
FemCalculixCantilever3D_newSolver.FCStd
158Kb
Schenkel.stp
576Kb
draft_test_objects.FCStd
FreeCAD Developers
117Kb
CAD
Abmessung
Blender
Datei Bearbeiten Rendern Fenster Hilfe
Layout Modeling Sculpting UV Editing Texture Paint Shading Animation
Objektmodus Ansicht Auswählen Hinzufügen Objekt
Benutzeransicht
(1) Collection | Cube
Szenensammlung
Collection
Camera
Cube
Light
3.6.2
Neue Datei
Allgemein
2D Animation
Sculpting
VFX
Video Editing
Leg los
Handbuch
Blender Website
Mitwirkende
Öffnen...
Letzte Sitzung wiederherstellen
Herausgabehinweise
Entwicklungsförderung
Transformieren
Delta-Transformation
IFC Derived Placements
Beziehungen
Sammlungen
Instancing
Bewegungspfade
Wiedergabe Keying Ansicht Markierung
Start 1 Ende 250
Auswählen
Ansicht drehen
Objekt Kontextmenü
Spyder (Python 3.11)
Debuggen Konsolen Projekte Werkzeuge Ansicht Hilfe
Quellcode Konsole Objekt
Gebrauch
Hier können Sie Hilfe zu jedem Objekt erhalten, indem Sie Ctrl+I davor drücken, entweder im Editor oder in der Konsole.
Hilfe kann auch automatisch angezeigt werden, nachdem eine linke Klammer neben ein Objekt geschrieben wurde. Sie können dieses Verhalten in Einstellungen > Hilfe aktivieren.
Hilfe Variablenmanager Abbildungen Dateien
Konsole 1/A
Python 3.11.2 (main, Mar 13 2023, 12:18:29) [GCC 12.2.0]
Type "copyright", "credits" or "license" for more information.
IPython 8.5.0 -- An enhanced Interactive Python.
In [1]:
IPython-Konsole Chronik
custom () Completions: custom LSP: Python Line 8, Col 1 UTF-8 LF RW Mem 23%