Erste Bekanntschaft mit Jupyter Notebook

Wir haben gestern die Arbeitsoberfläche installiert, mit der ihr Programmieren lernen werdet. Um sie zu startet beginnt ihr auf der Commandline. Ruft sie auf. Achtet darauf, dass ihr in eurem Virtualenvironment seit. Also z.B. eingeben `workon python3` und gebt dann ein `jupyter notebook` .

In eurem Browser sollte nun ein neues Fenster aufgehen. Falls es das nicht tut, öffnet den Browser und gebt ins URL-Feld ein: 127.0.0.1:8888. Nun sollte ein Fenster aufgehen mit Jupyter Notebook.

Warum **Jupyter Notebook?** Die Umgebung ist ideal, um mit dem Programmieren zu starten, weil man hier Code-Bausteine einzeln ausführen kann. Der Computer merkt sich im Hintergrund, was passiert ist. Ich arbeite noch immer damit, zumindest am Anfang. Und wenn mein Code fertig ist oder stabil genug, speichere ich ihn ab und löse ihn auf der Commanline aus. Das gute an Jupyter Notebook ist, dass man damit auch graphisch arbeiten kann.

Wenn wir Jupyter Notebook startet, sehen wir die Folderstruktur von der aus wir das Programm in der Commandline gesteuert haben. Wir können in Jupyter Notebook nun nicht mehr weiterrauf in der Folderstruktur. Sondern nur noch runter. Bevor man Jupyter Notebook startet, sollte man sich gut überlegen, wie man hin und her navigieren will.

Rechts haben wir die Möglichkeit, neue Files zu bauen. Klicken wir also auf "New" und dann auf "Python 3". Nun geht ein neuer Browser-Tab auf. Benennen wir es Übung5.

Jupyter Notebooks haben zwei Modi: MarkDown und Code. Wir können den Modus pro Zelle einstellen.

MarkDown

Den MarkDown-Modus wählen wir, um unseren Code zu beschreiben und zu strukturieren. Das ist sehr wichtig. Nur wirklich gut dokumenteirter Code ist auch nützlich - kann zu einem späteren Zeitpunkt wiederverwendet werden. Es lohnt sich deshalb früh anzueignen, Code möglichst genau zu dokumentieren.

Um auf MarkDown zu wechseln, gehen wir auf "Cell" und wählen "Markdown" an.

Um Text zu formatieren gibt es ein paar ganz grundlegende Befehle.

- Für Titel benutzt ihr #
- Für Untertitel ##, ### uns so weiter.
- Um Wörter einzufetten, rahmt ihr sie mit zwei Sternchen ein: **hallo**
- Aufzählungen könnt ihr mit Zahlen 1. oder mit einem simplen Bindestrich darstellen.
- Wer mehr wissen will, zum Beispiel Tabellen bauen usw., findet mehr hier.
- Wenn ihr Code-Zeilen nur beschreiben wollt, braucht ihr allerdings nicht in den Markdown-Modus zu wechseln. Es gibt dafür die einfache Möglichkeit, vor jeder Code-Zeile ein # zu setzen. Im Code-Modus macht das keinen Titel, sondern es sagt dem Computer, dass er alles was auf derselben Zeile dahinterkommt, nicht beachten muss.

Üben wir das ein wenig:

- Geht zurück zu Jupyter Notebook. Wechselt in der ersten Zelle in den Markdown-Modus.
- Gebt dort als Titel ein: "Mein erster Code"
- Drückt ausführen. Dann erfasst ihr als Text, wie im Markdown-Modus: "Als Erstes werde ich den den Text 'hello world' ausdrucken." Führt die Zelle wieder aus.
- Dann lässt ihr eine Zelle aus. Und gebt in der übernächten ein. "Hier verlange ich einen Input" ein.

Python 3

Wir arbeiten mit Python 3. Das ist die neueste Version. Ihr stösst im Netz oft auf Python 2. Der grösste Unterschied betrifft auf den ersten Blick das Print-Verhalten. Bei Python 3 müssen wir alles in Klammern verpacken, damit der Computer versteht, was wir wollen.

Also, geben wir ein `print("Hello World")`

So, und schon habt ihr euer erstes Programm geschrieben. Ihr seht, dass sich das statement **print** automatisch grün färbt. Das ist ein Zeichen dafür, dass der Computer das als Befehl erkannt habt. Und die Zeile, die der Computer ausspucken soll, ist in Anführungs- und Schlusszeichen. Das ist wichtig. Sie können doppelt oder einfach sein.

Alles was zwischen Anführungs und Schlusszeichen ist erkennt der Computer als Text. Er heisst **String**. Ihr werdet ab morgen sehr viel mehr über Strings kennenlernen. Die anderen Datenformate ins **Integers**, ganze Zahlen. Und **Floats**, Zahlen mit Dezimalstellen. Floats und Integers werden ohne Anführungs- und Schlusszeichen geschrieben. Wenn wir das tun, erkennt sie der Computer als Text.

Ihr erkennt schon daran, dass Zahlen für den Computer sehr viel wichtiger sind als Text. Für Text gibt es nur ein Format. Für Zahlen zwei.

Ich möchte hier nicht mehr weiter auf die verschiedenen Datentypen eingehen. Davon werdet ihr morgen noch genug erfahren.

Stattdessen möchte ich neben Print noch einen weiteren Befehl einführen: **input**. Diesen Befehl können wir verwenden, um mit Menschen zu inteagieren. Geben wir ein `Wie alt bist Du?` . Testet das.

Nun greife ich etwas vor und führe eine Variabel ein, auch zu Variabeln werdet ihr später viel mehr hören. Wir machen den Input zur Variabel. Also: `variabel = input("Wie alt nochmals?")` Wir können einer Variabel alles zuordnen, und er darf auch heissen wie er will. Nur Zahlen dürfen wir nicht verwenden. Sie sind ja bereits vergeben.

Als letztes führ ich die if-else-Clause. Auch dazu werdet ihr morgen mehr hören. Ich möchte euch hier nur eine kleine Ahnung geben, wir ihr künftig COde entwickelt und auslöst.

Jetzt speichert ihr das als .py ab, am besten auf den Desktop. Dort solltet ihr nun ein File mit dem Namen "Übung5.py" finden.

In der Commandline steuert ihr dieses File nun an, und löst es aus: Virtuel Environment starten. Und dann eingeben: `python Übung5.py` . Nun solltet ihr in der Commandline zu einem kleinen Hin- und Her mit eurem Computer haben.

Wenn ihr wissen wollt, wie der Jupyter Notebook Code genau aussieht, könnt ihr das bei Atom anschauen.