

Python-Programmierung im Informatikunterricht mit Jupyter Notebooks

Kutzera, J, Witte, L, Schunder, T.

DOI: 10.18420/ibis-03-01-05

Zusammenfassung

In diesem Artikel beleuchten wir die Einsatzmöglichkeiten JupyterLab-basierter Lernumgebungen im Informatikunterricht am Beispiel der Plattform Coding Labs. Zunächst geben wir einen Überblick über die Nutzung von JupyterLab im Bildungsbereich und stellen die Lernplattform Coding Labs vor. Anschließend beschreiben wir spezifische Gestaltungsmöglichkeiten für Lehrmaterialien innerhalb von JupyterLab. Abschließend fassen wir unsere Erkenntnisse aus Praxis-tests an Schulen zusammen. Dabei wurde deutlich, dass besonders die Möglichkeit der Kombination von theoretischen Unterrichtsinhalten mit praktischen Programmieraufgaben in JupyterLab-Umgebungen einen erheblichen Mehrwert bietet. Lehrkräfte hoben zudem positiv hervor, dass die geräteunabhängige Erreichbarkeit es den Schüler*innen erlaubt, Aufgaben und Mitschriften von zu Hause aus aufzurufen und zu bearbeiten.

Einleitung

Jupyter Notebooks stellen eine gute Möglichkeit dar, um Programmierwissen interaktiv und anwendungsorientiert an Lernende zu vermitteln. Dennoch werden sie von Lehrkräften im Schulalltag bislang nur selten verwendet. Im Forschungsprojekt Coding Labs wurde eine JupyterLab-basierte Plattform zum Programmieren Lernen entwickelt, die besonders für den Einsatz an Schulen und Universitäten geeignet ist.

In diesem Artikel stellen wir die Eigenschaften und Nutzungsmöglichkeiten von JupyterLab im Informatikunterricht am Beispiel der Coding Labs-Plattform vor. Ein besonderer Fokus liegt

dabei auf den Möglichkeiten bei der Erstellung und Gestaltung von Lehrmaterialien. So erhalten Lehrkräfte einen praxisnahen Einblick in die Funktionsweise von Jupyter Notebooks und können auf dieser Grundlage bewerten, ob die Nutzung von Coding Labs oder einer anderen JupyterLab-Umgebung eine sinnvolle Ergänzung für ihren Unterricht darstellt.

JupyterLab im Bildungsbereich

JupyterLab ist eine browserbasierte Umgebung zur Anzeige, Erstellung und Bearbeitung interaktiver Dokumente, sogenannter Jupyter Notebooks. Die JupyterLab-Umgebung erlaubt die gleichzeitige Nutzung mehrerer Notebooks in Tabs und enthält eine einfache Dateiverwaltung im Browser. Diese kann zum Hochladen und Organisieren von Notebooks und anderen Inhalten in einer Ordnerstruktur genutzt werden. Zudem unterstützt JupyterLab Login-Systeme, so dass Nutzer*innen individuelle Accounts mit persönlichem Dateiablageort erhalten. Die Umgebung kann direkt auf jupyter.org ohne Login ausprobiert werden¹. Neben Coding Labs² gibt es weitere Anbieter, die JupyterLab samt Login-System zur Verfügung stellen³. Darüber hinaus kann JupyterLab auch lokal gehostet werden. Ein Beispiel für die Ansicht einer JupyterLab-Umgebung befindet sich in Abbildung 1.

Jupyter Notebooks bestehen aus einzelnen Abschnitten, den sogenannten Zellen ("Cells"). Diese beinhalten entweder Programmcode, Medien oder in HTML oder Markdown formatierten Text

¹ <https://jupyter.org/try>

² <https://beta.codinglabs-projekt.de>

³ <https://hszg-jupyter.inf.hszg.de>

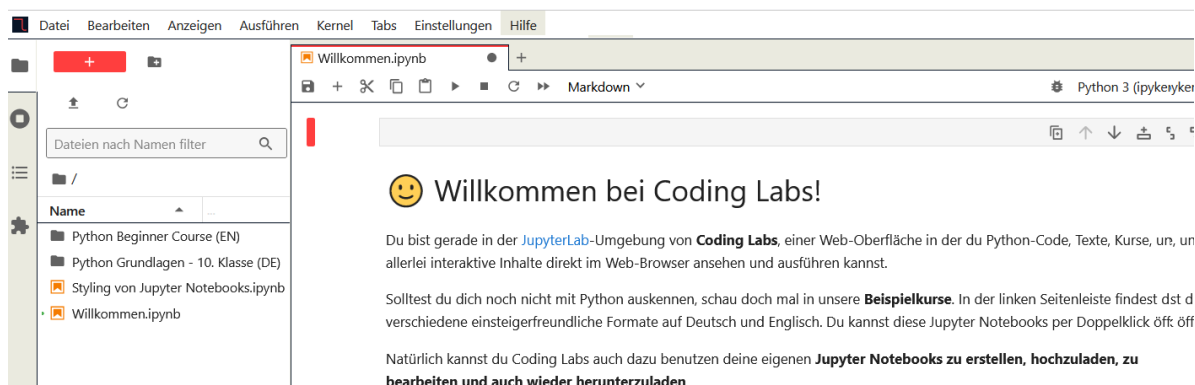


Abbildung 1: Die JupyterLab-Umgebung von Coding Labs

(Rule et al., 2018). Dank einer Vielzahl installierbarer Kernels unterstützt JupyterLab unterschiedliche Skript- und Programmiersprachen, wobei Python am häufigsten genutzt wird. Durch die Möglichkeit zur Kombination von Code und Text in den Notebooks eignen diese sich hervorragend für die Vermittlung von Coding Skills. Dabei können sie gleichzeitig als Lehr- und Übungsmaterial sowie als Programmierumgebung und Mitschrift für die Lernenden fungieren.

Dennoch sind Jupyter-basierte Umgebungen im Bildungs- und insbesondere im Schulbereich bislang nur wenig verbreitet. Mögliche Gründe sind hier mangelnde Bekanntheit oder Unsicherheiten hinsichtlich der technischen Implementierung. Auch rechtliche Bedenken, insbesondere bezüglich der DSGVO-Konformität, stellen eine potenzielle Hürde dar (Kutzera et al., 2023). Dennoch wurde der Einsatz von Jupyter Notebooks im Schulbereich bereits mehrfach beschrieben, beispielsweise in einem Praxisbericht zu einem Kurs für Datenanalyse im Informatikunterricht (Podworny et al., 2022). Bovermann (2024) stellt zwei Bibliotheken für die Darstellung von Graphen in Jupyter Notebooks vor und erläutert deren Potenzial für den Informatikunterricht. Eine weitere Ausarbeitung diskutiert den Einsatz von Gamification-Ansätzen in JupyterLab zur Förderung des Lernprozesses (Brocker et al., 2022).

Die Plattform Coding Labs

Coding Labs ist eine kostenlose, browserbasierte Plattform zum Programmieren mit Python, die auf JupyterLab basiert. Nach dem Einloggen steht den Nutzenden eine persönliche JupyterLab-Umgebung samt Dateiverwaltung zur Verfügung. Das Account-System erlaubt es ihnen, von verschiedenen Endgeräten aus in ihrer eigenen Programmierumgebung zu arbeiten, Änderungen zu speichern und ihre Arbeit jederzeit fortzusetzen. Die Plattform wurde speziell für den Einsatz an Schulen und Universitäten sowie für lebenslanges Lernen entwickelt. Sie wird in Deutschland gehostet und ist DSGVO-konform, was insbesondere für den Einsatz in der Schule wichtig ist.

Der Einsatz von JupyterLab im Informatikunterricht

Der Einsatz von JupyterLab-Umgebungen im Informatikunterricht ist besonders dann empfehlenswert, wenn Inhalte mithilfe textbasierter Programmiersprachen vermittelt werden sollen. Bezogen auf das Kompetenzmodell des Bildungsstandards Informatik für die Sekundarstufe II der Gesellschaft für Informatik (GI) e.V. (Röhner 2016),

ist die Nutzung von JupyterLab besonders für die Lehre der Inhaltsbereiche „Algorithmen“ und „Informationen und Daten“ geeignet. Für andere Inhaltsbereiche wie „Sprachen und Automaten“ kann die Umgebung zusätzlich um spezielle Bibliotheken wie ipyturtle erweitert werden.

Erstellung von Lehrmaterial: wieviel Zeit steht der Lehrkraft zur Verfügung?

JupyterLab-basierte Plattformen lassen sich mit wenig Aufwand für den Unterricht vorbereiten. Allerdings erfordert das Gestalten von passendem Lehrmaterial einen gewissen Arbeitsaufwand. Lehrkräfte, die bereits Materialien in Form von Jupyter Notebooks zur Verfügung haben, können diese direkt verwenden. Lehrmaterialien in anderen Formaten (z. B. HTML- oder Textdateien) müssen hingegen manuell in Jupyter Notebooks übertragen und ggf. angepasst werden.

Für Lehrkräfte mit wenig Einarbeitungszeit gibt es online eine wachsende Auswahl von OERs (Open Educational Resources) im Jupyter Notebook-Format. Allerdings sollte Zeit für die Suche, Sichtung und Anpassung des Materials an den Lehrplan und den Kenntnisstand der Schüler*innen eingeplant werden. Auch im Rahmen des Coding Labs-Projekts wurde frei verfügbares und individuell anpassbares Kursmaterial erstellt, welches sich für den Einstieg in Python eignet⁴.

Wer sich intensiver mit der Erstellung von Lehrinhalten für JupyterLab-Plattformen auseinandersetzen möchte, hat durch die Kombination von Markdown- und Python-Zellen in den Jupyter Notebooks eine Vielzahl gestalterischer Möglichkeiten. Diese werden in den folgenden Abschnitten näher erläutert.

Jupyter Notebooks ansehnlich gestalten: Möglichkeiten innerhalb der Markdown-Zellen

Jupyter Notebooks ermöglichen die Formatierung von Inhalten mithilfe von Markdown und vereinfachtem HTML. Dadurch lassen sich u.a. Überschriftenhierarchien, farbliche Hervorhebungen und Strukturelemente wie Trennlinien und visuell abgesetzte Boxen erzeugen. Grafiken können entweder per Drag-and-Drop direkt ins Notebook eingefügt oder über das HTML-Element `` eingebunden werden. Während eingebettete Bilder fest im Notebook-Quelltext eingebunden werden, sodass sie nicht getrennt gespeichert werden müssen, können über `` eingebundene Grafiken flexibel skaliert werden.

⁴ <https://github.com/KMI-KPZ/Coding-Labs-Lerninhalte>

Zusätzlich gibt es weniger bekannte Gestaltungsmöglichkeiten, die jedoch besonders für den Unterrichtseinsatz relevant sind. Mit HTML `<details>` Elementen lassen sich Notebook-Bereiche ausblenden, welche erst durch Anklicken sichtbar werden. So können beispielsweise Aufgabenlösungen verborgen werden. Zur abwechslungsreicheren Gestaltung von Text und Überschriften können Unicode-Emojis verwendet werden. HTML `` Elemente können zum Erzeugen von gesondert formatierten Boxen und Feldern verwendet werden. Die Elemente können farblich hervorgehoben und als Container für Exkurse und Tipps genutzt werden. Außerdem erlaubt das Einbinden von LaTeX-Code die Verwendung mathematischer Formeln. Alle Gestaltungsmöglichkeiten sind in Abbildung 2 noch einmal aufgezeigt und kurz erklärt.

Gestalten von Code-Aufgaben und Tools: Möglichkeiten innerhalb der Python-Zellen

Bei der Erstellung von Programmieraufgaben können die Code-Zellen entweder leer gelassen werden oder bereits Code enthalten, den die Schüler*innen vervollständigen müssen. Aufgabenanweisungen können in Form von Python-Kommentare eingefügt werden, z. B. «`## Ergänze hier deinen Code`». Da Code-Zellen versehentlich gelöscht oder verändert werden können, muss dabei sichergestellt sein, dass der existierende Code wiederhergestellt werden kann. Eine Möglichkeit ist ein über der Code-Zelle platziertes `<details>`-Element mit einer Beschriftung wie «Code zum Wiederherstellen anzeigen», welches den Code nach Anklicken anzeigt, so dass die Schüler*innen den ursprünglichen Code bei Bedarf wieder in die Zelle kopieren können.

Eine weitere wichtige Funktion von Jupyter Notebooks ist hier zudem das Sperren von Zellen. Für jede Zelle (sowohl Python als auch Markdown) kann festgelegt werden, ob sie editierbar sein soll oder nicht. Dies ist insbesondere bei Python-Zellen mit vorgegebenem Übungscode wichtig, sofern der Code nicht verändert werden soll.

JupyterLab enthält zudem umfangreiche Möglichkeiten zur Visualisierung und Interaktion. Für die Visualisierung mathematischer Funktionen und für das Plotten von Ergebnissen aus Datenanalysen hat sich z. B. die Bibliothek `Matplotlib`⁵ bewährt.

Soll ein Spiel oder eine interaktive Anwendung entwickelt werden, eignen sich insbesondere die Bibliotheken `ipycanvas` und `ipywidgets`. `ipycanvas` gewährt Zugriff auf die Canvas-API im Browser und erlaubt das direkte Zeichnen von Pixeln, Linien und Formen. Dabei sind verschiedene Darstellungsformen möglich, von Retro-Pixelgrafiken bis hin zu Skizzenoptik. Zudem können PNG-Grafiken auf dem Canvas eingebettet und animiert werden.

Mit der Library `ipywidgets` steht eine große Auswahl interaktiver Elemente für die Eingabe und Ausgabe zur Verfügung, z.B. Buttons, Sliders, Checkboxen oder Labels. Diese sind mit Python-Funktionen verknüpft, die immer dann ausgeführt werden, wenn auf ein Element geklickt oder ein Slider bewegt wird. Darüber hinaus lassen sich diese Elemente gezielt je nach Bedarf manipulieren, z. B. durch Ein- oder Ausblenden, oder durch Deaktivieren.

Weiterhin ist über die Code-Zellen auch die Einbindung von YouTube-Videos möglich. Diese werden angezeigt, sobald die Code-Zelle ausgeführt wurde. Der entsprechende Code ist in Ab-

⁵ <https://matplotlib.org>



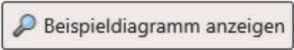
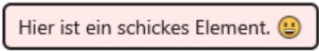
Designelement	Code	Details	Ziel
Unicode-Emojis  Willkommen	Markdown	Direktes Einfügen von Unicode-Übersichtsseiten	Elegante und einfache Methode, um z. B. Überschriften ansprechend zu gestalten
Grafik (PNG, JPG) 	Markdown, HTML	Einbindung durch Drag+Drop oder Verlinkung via HTML <code></code> tag	Grafiken können Teil der Lehrmaterialien aber auch Übungsaufgabe sein, da Schüler*innen Grafiken einfach selbst einfügen können
HTML <code><details></code> Element 	HTML	Verstecken von Hinweisen und Lösungen, die erst nach Klicken auf Button sichtbar sein sollen	Schüler*innen bekommen die Gelegenheit, erst selbst nach der Lösung zu suchen, bevor Hinweise angezeigt werden
HTML <code></code> Element 	HTML	Erstellen von grafisch ansprechenden und flexibel konfigurierbaren Designelementen	Auflockerung des Lehrmaterials, Schaffen von Abwechslung und Strukturierung
Mathematische Formeln $f(x) = \int_{-\infty}^{\infty} e^{-x^2} dx$	Markdown	Markdown unterstützt die Einbindung von LaTeX-Code	Verwendung der mathematischen Notationen, wie sie auch im Mathematikunterricht Einsatz finden

Abbildung 2: Markdown-basierte Gestaltungselemente und ihre Anwendungsbereiche (eigene Darstellung)

bildung 3 zu finden, die eine Übersicht der auf Python basierenden Gestaltungselemente bietet.

Im Projekt Coding Labs wurden verschiedene Libraries speziell für Lehrkräfte entwickelt, die bei der Erstellung eigener Lernmaterialien verwendet und angepasst werden können. Die Libraries sind auf GitHub verfügbar⁶.

So erlaubt die Library CL_canvas die Verwendung von ipycanvas-Funktionen mit weniger Code, so dass der Einstieg in das Handling von Grafik und Buttons mit ipycanvas vereinfacht wird. Zusätzlich wurde die Library sherlock.holmes entwickelt, die es Lehrkräften erlaubt, in ihre Übungen detaillierte Code-Überprüfungen zu integrieren, die von den Schüler*innen selbstständig ausgeführt werden können. So entsteht ein motivierendes Moment, da die Schüler*innen direktes Feedback darüber erhalten, welche Teile ihres Codes bereits funktionieren und welche noch verbessert werden müssen. Die Library drwatson erlaubt das Einbinden von kleinen Multiple-Choice Quizen in die Übungsaufgaben, mit denen die Schüler*innen ihr Wissen selbst testen können. Im GitHub-Repository befinden sich detaillierte Anleitungen zur Verwendung der hier genannten Tools, sowie eine Übersicht zu den in den vorherigen Absätzen beschriebenen gestalterischen Möglichkeiten.

Erfahrungen aus der Praxis

Während der Entwicklung der Coding Labs-Plattform wurde diese von einigen Lehrkräften im Unterricht getestet. Die Erkenntnisse aus diesen Tests sind nicht nur für Coding Labs, sondern für alle JupyterLab-Plattformen relevant.

Allgemein kommen Schüler*innen gut mit JupyterLab-Plattformen und Jupyter Notebooks zurecht. Von den Lehrkräften wird besonders geschätzt, dass die Jupyter Notebooks gleichzeitig als Lehrmaterial und persönliche Mitschrift für die Schüler*innen fungieren. So kann neu erlangtes Theoriewissen in den ausführbaren Code-Zellen sofort praktisch angewendet werden. Zudem können die Kapitel des Lehrmaterials problemlos aufeinander aufbauend gestaltet werden, da Code aus vorherigen Kapiteln für die Schüler*innen einfach aufrufbar bleibt und somit direkt wiederverwendet werden kann.

Zusätzlich ermöglicht der browserbasierte Zugang der Lehrkraft, den Schüler*innen direkt auf der Plattform zu lösende Hausaufgaben aufzugeben. Die bearbeiteten Aufgaben können in der nächsten Unterrichtsstunde ohne weitere Zwischenschritte vom Schul-PC aus aufgerufen werden.

Ein genereller Nachteil der Plattformen ist die Abhängigkeit von einer stabilen Internetverbindung, da JupyterLab-Umgebungen sonst nicht zuverlässig funktionieren. Sofern nicht lokal gehostet, ist ein „Offline“ Arbeiten, z. B. im Rahmen von Kurzkontrollen, nicht möglich.

⁶ <https://github.com/KMI-KPZ/Coding-Labs-Lerninhalte>

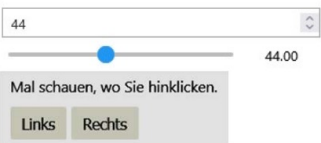
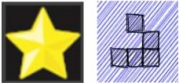

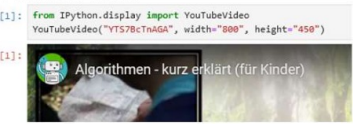
Designelement	Library	Details	Ziel
Interaktive-Elemente 	ipywidgets	Ipywidgets sind mit definierbaren Funktionen verbunden, welche bei Aktivierung der Elemente ausgeführt werden	Slider, Buttons, Radiobuttons und andere interaktive Elemente als Nutzereingabe und -Ausgabe für Pythonprogramme
Bilder und Pixelgrafiken 	ipycanvas	Erstellen, Anzeigen, Ausblenden und Animieren von Grafiken und einzelnen Pixeln	Motivierende Übungsaufgaben, z. B. Erstellen grafisch ansprechender Spiele
Turtle-Grafiken 	ipyturtle	Viele Libraries, z. B. Turtle, existieren in einer Jupyter-kompatiblen Variante	Verwenden von etablierten Lernmitteln für Themen wie Automaten
YouTube-Video 	ipython	Codzeile muss zum Anzeigen einmalig ausgeführt werden	Einbindung von weiterführenden Inhalten in Videoform

Abbildung 3: Python-basierte Gestaltungselemente und ihre Anwendungsbereiche (eigene Darstellung)

Fazit und Ausblick

JupyterLab-Plattformen wie Coding Labs stellen ein geeignetes Werkzeug für das Erlernen der Python-Programmierung und die Lehre der Themen Algorithmen und Datenstrukturen im Informatikunterricht dar. Durch das Account-System erhalten Schüler*innen eine persönliche JupyterLab-Umgebung. Dort sind sowohl Lehrmaterial als auch Übungen und Mitschriften stets verfügbar und können im Unterricht und zu Hause eingesehen und bearbeitet werden.

Das Erstellen von Lehrmaterial für JupyterLab-basierte Plattformen ist mit einem gewissen Zeitaufwand verbunden. Allerdings kann bereits in anderen textbasierten Formaten existierendes Material einfach an die Plattform angepasst werden und durch die Möglichkeiten, die Jupyter Notebooks bieten, z. B. die direkt ausführbaren Code-Zellen, sinnvoll erweitert werden. Einmal für die Nutzung in JupyterLab angepasstes Material kann einfach verteilt, wiederverwendet und auf allen JupyterLab-basierten Plattformen genutzt werden.

Auch jenseits des Informatikunterrichtes ist ein Einsatz von JupyterLab-basierten Plattformen denkbar: So könnten im Mathematikunterricht mithilfe der Bibliothek Matplotlib interaktive Visualisierungen mathematischer Funktionen erstellt werden, bei denen die Schüler*innen Variablen über Schieberegler anpassen können.

Somit ermöglichen JupyterLab-Plattformen eine nahtlose Verknüpfung von theoretischem Wissen und praktischer Anwendung, indem sie beides in einer interaktiven Umgebung vereinen und jederzeit zugänglich machen – sowohl im Informatikunterricht als auch in anderen Fachbereichen.

Förderhinweis

Das Projekt Coding Labs wird im Rahmen der Maßnahme „Initiative Nationale Bildungsplattform“ (Förderkennzeichen 16INB2008 ff.) vom Bundesministerium für Bildung und Forschung (BMBF) gefördert und vom Projektträger VDI-VDE IT betreut. Die Verantwortung für den Inhalt dieser Veröffentlichung liegt bei den Autor*innen.

Quellen

Alle Webseiten/Links wurden zuletzt geprüft am 28.11.2024.

Bovermann, Klaus (2024): Graphen im Unterricht Mit Hilfe von Python und Jupyter-Notebooks, IBIS Jg. 2 (2024), Nr. 1

Brocker, Annabell; Judel, Sven; Roepke, Rene; Mihailovska, Nikol und Schroeder, Ulrik (2022): Gamifying JupyterLab to Encourage Continuous Interaction in Programming Education, in Kiili et al.: Games and Learning Alliance, GALA 2022, Lecture Notes in Computer Science, vol 13647

Kutzera, Joachim; Wöhlert, Romy; Friedrich, Julia; Römer, Vanita und Richter, Patrick (2023): Future Learning in a Collaborative "Laboratory" Environment - Requirements to Build Up Future Skills, in Camarinha-Matos et al.: Collaborative Networks in Digitalization and Society 5.0. PRO-VE 2023. IFIP Advances in Information and Communication Technology, vol 688

Podworny, Susanne; Hüsing, Sven und Schulte, Carsten (2022): A PLACE FOR A DATA SCIENCE INTRODUCTION IN SCHOOL: BETWEEN STATISTICS AND PROGRAMMING, Statistics Education Research Journal (IASE/ISI), 21(2), Article 6

Röhner, Gerhard (2016): Bildungsstandards Informatik für die Sekundarstufe. Beilage zu LOG IN, 36. Jg. (2016), Heft Nr. 183/184.

Rule, Adam; Tabard, Aurélien und Hollan, James D. (2018): Exploration and Explanation in Computational Notebooks, CHI 2018, April 21–26, 2018, Montreal, QC, Canada

Lizenz



Dieser Artikel steht unter der Lizenz CC BY NC SA 4.0 zur Verfügung.

Kontakt

Joachim Kutzera, Leonie Witte, Toni Schunder
 Institut für Angewandte Informatik (InfAI) e.V.
 Leipzig
 kutzera@infai.org