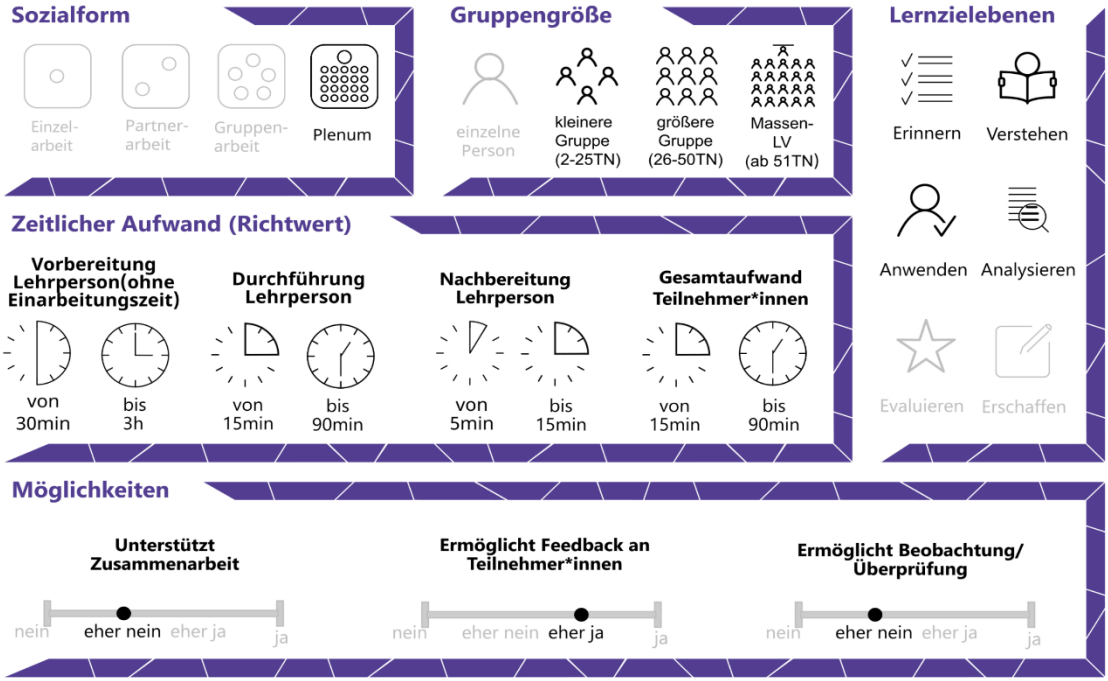
Programmcode präsentieren

Stand: 27.05.2024

Kurzbeschreibung

Programmcode zu präsentieren ist ein wesentlicher Bestandteil in der Programmierlehre und den Datenwissenschaften. Damit die Teilnehmer*innen (TN) dem gezeigten Programmcode folgen können und motiviert bei der Sache bleiben, sind gewisse Vorkehrungen und Vorbereitungen zu treffen. In dieser Form der Lehre sind digitale Hilfsmittel wesentlicher Bestandteil, da der Code selbst digitaler Natur ist. Dieser Use Case stellt verschiedene Formen, Methoden und Tools vor, die das Präsentieren von Programmcode unterstützen, um den Lernerfolg zu maximieren.

Allgemeine Eckdaten



 Aktuelle Version: 30.10.2019

Inhaltsverzeichnis

Gründe für den Einsatz

- Das Verständnis von Programmcode ist ein wesentlicher Bestandteil in programmier- und daten-fokussierten Studien.
- Programmcode bietet eine Schnittstelle zur Verknüpfung von Theorie mit Praxis (z.B. Vorlesung und Übung).
- Das Herzeigen von Programmcode führt zu einem besseren Verständnis von Konzepten in der Programmierlehre (z.B. bei Datenstrukturen).
- Die in diesem Use Case beschriebenen Methoden und Tools eignen sich gut für Aufzeichnungen und Livestreams bei großen Teilnehmer*innensettings.
- Wenn Programmcodebeispiele während der Präsentation ausgeführt werden, ermöglicht dies den Studierenden, Erfahrung mit Entwicklungsumgebungen zu sammeln.

Technische Infrastruktur / Empfehlungen

Für das Präsentieren von Programmcode sind ein Computer und Infrastruktur zur Übertragung (Beamer oder Streaming Software) wesentlich. Sollte die Präsentation gestreamt werden, sind weitere Geräte wie Kamera, Mikrofon und eventuell ein eigener Streaming PC von Vorteil. Weiters kann ein eigenes Tablet zum Zeichnen nützlich sein. Zur Vorbereitung und Darstellen vom Programmcode sind gängige Entwicklungsumgebungen (alternativ zum Texteditor) von Vorteil.

Rolle der Lehrperson

Die Lehrperson (LP) ist typischerweise der*die Wissensvermittler*in und steht im Zentrum der Lerneinheit. Jedoch ist ebenso die Partizipation der TN ein wesentliches Element, damit sie den gezeigten Programmcode besser verstehen bzw. selbst nachprogrammieren können. Manche Lerneinheiten ermöglichen es den TN, selbst am Laptop mitzuarbeiten oder der LP vorgeschlagene Adaptierungen des Programmcodes live einzubauen und sofort auszuführen.

Einsatzmöglichkeiten / Methoden

Das Präsentieren von Programmcode kann in verschiedenen Szenarien eingesetzt werden, wobei die Anzahl der TN bei der Auswahl des Szenarios wesentlich ist:

Im Hörsaal

Das Präsentieren im Hörsaal ermöglicht es den TN, unmittelbar Fragen (zum Verständnis oder nach weiterführenden Informationen) an die LP zu stellen. Umgekehrt bekommt die LP ein Gefühl, wie gut der gezeigte Stoff von den TN verstanden wird und kann dialogische Partizipation anregen.

Videoaufzeichnung

Das Aufzeichnen von Lehrvideos (entweder separat oder während einer Lerneinheit) erlaubt es den TN, den Stoff in eigener Geschwindigkeit durchzugehen. So können erfahrende TN bei erhöhter Abspielgeschwindigkeit das Gelernte verfestigen, während unerfahrene TN das Video pausieren können, um das Gezeigte selbst auszuprobieren. Die Videoaufzeichnung hat einen großen Wiederverwendungsfaktor.

Online-Stream

Mit dem Online-Streaming der Lerneinheit kann eine sehr große Anzahl an TN gleichzeitig erreicht werden, wobei den TN trotzdem eine Interaktionsmöglichkeit mittels Chats ermöglicht werden kann. Bei großer Anzahl an TN werden ein oder mehrere Chatbeauftragte benötigt, die das Bindeglied zwischen LP und TN darstellen. Chatbeauftrage haben die Aufgabe, den Chatverlauf zu überwachen, einfache Fragen direkt zu beantworten, relevante Fragen an die LP weiterzugeben und allenfalls als Moderator*in einzugreifen (z.B. Löschen unpassender Nachrichten oder Muten von TN). Neben dem Chat selbst kann eine Interaktion mit Umfragetools oder Online-Quizzes ermöglicht werden.

Live Coding

Im Live Coding kann ein Programm zusammen mit den TN sukzessive erarbeitet werden. Das Live Coding ermöglicht daher eine hohe Interaktion zwischen LP und TN in kleinen oder mittelgroßen Gruppen. Anders als bei den anderen Szenarien ist beim Live Coding das Endprodukt nicht im Vorhinein absehbar. Trotzdem ist

das Vorbereiten eines Planes zur groben Strukturierung der Einheit empfehlenswert. Weiters empfiehlt es sich, bewusst Fehler einzubauen, die von den TN erkannt und aufgezeigt werden sollen, um die Partizipation zu erhöhen.

Zeitlicher Aufwand

Da beim Präsentieren von Programmcode dieser zuerst ausgearbeitet werden muss, kommt es anfangs zu einem Mehraufwand. Dieser Mehraufwand kann jedoch reduziert werden, in dem bereits vorhandener Programmcode als Basis genommen wird. Bei sich wiederholenden Lerneinheiten ist der langfristige Aufwand geringer, jedoch muss der Programmcode gegebenenfalls gewartet werden, um nicht zu veralten (z.B. Anpassen der verwendeten Programmbibliotheken).

Tipps zur Umsetzung

- Damit der Programmcode gut leserlich ist, muss auf Schriftgröße und Kontrast geachtet werden. Die Standardeinstellungen von Entwicklungsumgebungen eignen sich typischerweise nicht zur Präsentation, da sie den Code eher klein darstellen. Einige Entwicklungsumgebungen beinhalten einen Präsentationsmodus, wo die Schriftgröße automatisch vergrößert wird und unnötige Elemente ausgeblendet werden.
- Zur Unterstützung der Erklärung des Programmcodes empfiehlt es sich, die Zeilennummern anzuzeigen und die verschiedenen Elemente farblich zu markieren (= Code Highlighting; Anmerkung: wird von den meisten Entwicklungsumgebungen automatisch angewandt).
- Neben der Präsentation des Programmcodes selbst wird empfohlen, benötigte Tools ebenfalls herzuzeigen. Zum Beispiel kann für das Kompilieren vom Programmcode das Terminal, wo der Befehl ausgeführt wird, hergezeigt werden.
- Es empfiehlt sich, den TN den Programmcode in der Ursprungsform zur Verfügung zu stellen. Hierfür eignet sich besonders ein öffentliches Repositorium, wo der Programmcode hochgeladen und dann in der Präsentation oder auf der Kurswebseite verlinkt wird.

Vorteile / Herausforderungen

- Das Präsentieren von Programmcode stellt eine sehr anwendungsbezogene Art des Frontalunterrichts dar, da ein ausführbarer Programmcode ein fertiges Endprodukt darstellt. Daher ist die Barriere zwischen Vortrag und Anwendung geringer.
- Die Präsentation von Programmcode kann als Hands-on Einheit durchgeführt werden, wo TN aktiv mitarbeiten und mitgestalten können.
- Während der Präsentation kann der Lernfortschritt und -erfolg mittels kurzer Beispiele für die TN (z.B. Code-Quizzes) überprüft werden und ermöglicht so einen besseren Überblick für die LP.
- Die Motivation der TN ist ein wesentlicher Faktor für den Erfolg, da sobald die TN den Anschluss verlieren, sie häufig auch die Syntax nicht mehr verstehen können und damit der Nutzen der Lerneinheit für sie stark reduziert ist.
- Programmierlehrveranstaltungen werden oft von vielen TN gleichzeitig besucht (Massen-LV). Das Präsentieren von Programmcode eignet sich gut für große Anzahl an TN, aber bringt eigene Herausforderungen mit sich, um die Interaktion zwischen TN und LP zu ermöglichen.
- Da typischerweise die Zeit fehlt, um ein Beispiel umfänglich durchzuarbeiten, werden oft Programmteile ausgeblendet. Dies kann zu Verständnisproblemen der TN führen und sollte daher mit Bedacht eingesetzt werden. Jedenfalls soll der ausgeblendete Programmcode den TN anderweitig (z.B. durch Verlinkung) zur Verfügung gestellt werden.

Einfluss auf Lernerfolg

Für den Lernerfolg der TN ist es maßgeblich, dass sie sich aktiv mit dem gezeigten Programmcode auseinandersetzen. Ein Faktor, der dabei hilfreich sein kann, ist das Bereitstellen des gezeigten Programmcodes durch die LP. Während sich Programmcode schwer in einem klassischen Skriptum darstellen lässt, bietet gut strukturierter und kommentierter Programmcode zusammen mit anderen Lernmaterialien eine solide Lernbasis. Am Ende hängt der erfolgreiche Lernprozess jedoch überwiegend von der Selbständigkeit der TN ab.

Einfluss auf Motivation

Das Präsentieren von Programmcode kann sehr schnell monoton erscheinen, wenn die TN passiv sind. Um die Motivation der TN zu erhalten, ist die Interaktivität wesentlich. Je nach Lerneinheit kann es förderlich sein, die TN selbst mittippen zu lassen. Diese Form der Interaktivität sollte aber mit Bedacht gewählt werden und eignet sich weniger für tiefergehendes Verständnis.

Rechtliche Aspekte

Mit diesem Absatz möchten wir Sie für rechtliche Aspekte beim Einsatz von digitalen Technologien in Unterricht und Lehre sensibilisieren. Gesetzliche Bestimmungen sind jedenfalls einzuhalten. Für diesen Use Case sind insbesondere folgende Rechtsthematiken relevant:

- Urheberrecht (Programmcode unterliegt dem Urheberrecht)
- Nutzungsbedingungen (von Tools wie Entwicklungsumgebungen, aber auch teilweise von Programmiersprachen selbst)
- Datenschutzgrundverordnung (inkl. Datensicherheit)

Bitte wenden Sie sich bei weiteren Fragen an die zuständige Abteilung(en) Ihrer Institution.

Mögliche Tools für Umsetzung

Entwicklungsumgebungen und Dokumentation

Die Präsentation von Programmcode ist immer mit dessen Erstellung verbunden. Dabei sind dafür benötigten Entwicklungsumgebungen und die Dokumentation des Programmcodes (sowie die Versionierung) wesentlich. Diese können entweder ausschließlich im Hintergrund verwendet werden oder explizit in der Einheit gezeigt werden.

- Visual Studio Code (VS Code) ist eine gängige und freie Entwicklungsumgebung, die sich unabhängig von der Programmiersprache einsetzen lässt. Durch die einfache Oberfläche ist VS Code gut für Programmieranfänger*innen geeignet. Da sich VS Code mit Erweiterungen beliebig anpassen lässt, wird sie auch gerne von erfahrenen Programmierer*innen verwendet. Die Kombination dieser Eigenschaften führt dazu, dass VS Code sich ideal in die Lehre integrieren lässt.

- Das Jupyter Ecosystem eignet sich gut für die Datenwissenschaften mit Fokus auf Programmiersprachen für wissenschaftliche Berechnungen, wie Python, R oder Julia. Es gibt mit Jupyter Notebook und JupyterLab zwei web-basierte Oberflächen zur interaktiven Entwicklung so genannter computergestützter Notebooks. Dabei werden Programmcode, Visualisierungen und Text in linearer Form verwoben, was die Lesbarkeit und das Verständnis des Programmcodes unterstützt. Die damit erstellten Notebooks eignen sich ideal zur Anwendung in der Lehre, wo neben dem Programmcode selbst andere Aspekte wie Datenauswertungen eine wesentliche Rolle spielen.
- Eine Markdown Dokumentation zum Beispiel in Notion kann sehr nützlich sein. Markdown ist eine einfache Sprache zur Erstellung von formatiertem Text, welche von Notion, einer populären Plattform zur Erstellung von Dokumentationen und Wikis, unterstützt wird. Da in Markdown ein Blocktyp für Code Snippets (kleine, nicht selbstständig ausführbare Programmteile) inkludiert ist, eignet sich dies besonders zur einfachen Erstellung einer Kurswebseite für weiterführende Materialien.
- GitHub ist eine vielgenutzte Plattform, die basierend auf Git die Versionierung von Programmcode ermöglicht. Dies ist unentbehrlich, wenn der Programmcode einen längeren Lebenszyklus hat und damit regelmäßig gewartet werden muss. Updates zum Programmcode oder dessen Dokumentation können damit klar ersichtlich gemacht werden.

Präsentationstools

Die Wahl der passenden Werkzeuge zur Präsentation ist ein weiterer wesentlicher Faktor, welcher von der Präsentationsform und dem Programmcode selbst abhängig ist. Dabei sind die Darstellung von Programmcode (und Code Highlighting), sowie das Wechseln zwischen Programmcode und anderen Modalitäten essenziell.

- Google Slides erlaubt ein kollaboratives Erstellen von Folien, welche direkt per Link geteilt werden können. Es ist damit eine Alternative zu PowerPoint und anderer bekannter Präsentationsoftware. Um Code Snippets besser darzustellen, kann auf Hilfswerkzeuge wie den SlidesCodeHighlighter zurückgegriffen werden, wodurch Code automatisch farbliche Unterscheidungen (ähnlich zu Entwicklungsumgebungen) bekommt.

- LaTeX erlaubt die Erstellung von Folien durch die Benutzung von spezialisierten Klassen, wie etwa beamer. Da LaTeX sehr flexibel und erweiterbar ist, kann Code einfach dargestellt werden, zum Beispiel mit dem minted package, welches Code Highlighting ermöglicht.
- Reveal.js basiert auf HTML zur Erstellung und Wiedergabe von Präsentationen. Reveal.js unterstützt diverse Arten von Medien, wobei das Präsentieren von Programmcode direkt unterstützt wird. Da Reveal.js auf Webtechnologien basiert, ist dieses Tool für Programmierer*innen von Vorteil, da diverse Funktionen mit bereits bestehenden Sprachen umgesetzt werden können. Zum Beispiel basiert das Erstellen und die Benützung eines Präsentations-Designs auf CSS, während Animationen auf JavaScript basieren. Dies hilft ebenso bei der Präsentation des Programmcodes, wo Code Highlighting mit CSS konfiguriert wird und man u.a. mit JavaScript Codeteile hervorheben und animieren kann.
- Quarto ermöglicht die Konvertierung von Notebook Formaten (z.B. Jupyter) zu anderen Arten von Dokumenten, wie eben auch zu Präsentationen. Dabei ist die Auswahl, ob die Endpräsentation zu PowerPoint, LaTeX Beamer oder Reveal.js umgewandelt wird, freigestellt. Die Umwandlung wird mittels YAML konfiguriert, entweder global oder auf Codezellen-Ebene. Damit lässt sich ein normal ausführbares Programm direkt in eine Präsentation umwandeln.
- Die Open Broadcasting Software (OBS) ist eine freie Anwendung, die sich zur Aufzeichnung und Live-Übertragung der Präsentation eignet. Die Anwendung ist sehr funktionsreich und unterstützt unter anderem das nahtlose Wechseln zwischen verschiedenen Ansichten. Da beim Programmieren oft von mehreren Anwendungen gebraucht gemacht wird (z.B. Entwicklungsumgebung, Terminal mit Compiler) ist diese Funktion von zentraler Bedeutung.

Schnittstellen und Interaktion

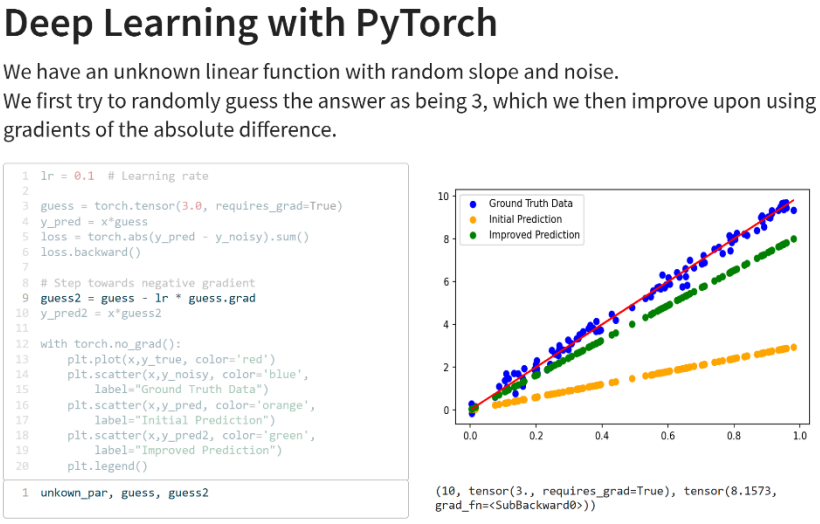
Neben der Präsentation selbst ist die Schnittstelle zwischen LP und TN ausschlaggebend für den Erfolg. Neben den klassischen Interaktionen in der Einheit selbst können die folgenden Schnittstellen/Tools ebenfalls die Interaktion fördern:

- Mentimeter ist eine Feedback-App, die vielfältige Möglichkeiten bietet, um mit TN in Interaktion zu treten und eignet sich gut für größere Gruppen. Es können damit sowohl Umfragen als auch Quizzes in Echtzeit umgesetzt werden.

- Virtuelle Maschinen (VMs), z.B. mittels VirtualBox, ermöglichen es LP und TN, den Programmcode unter gleichen Bedingungen (= virtuelle Umgebung) auszuführen. Damit kann die LP den TN eine vorab korrekt konfigurierte Umgebung zur Verfügung stellen, um typische Probleme, die aufgrund unterschiedlicher Systeme entstehen, zu vermeiden. VMs eignen sich besonders in Übungsszenarien oder für Hausaufgaben.
- YouTube ist eine Videoplattform, auf der Aufzeichnungen hochgeladen aber auch live gestreamt werden können. YouTube hat mehrere Funktionen, die Interaktion ermöglichen, wie die Kommentarfunktion bei Videos oder aber auch der Live-Chat bei einem Stream. Durch das Veröffentlichen der Einheit auf YouTube kann eine größere Anzahl an TN erreicht werden.

Anwendungsbeispiel

In einer Vorlesung zum Thema Angewandte Künstliche Intelligenz (KI) des Studiengangs Computer Science mit etwa 100 TN wird Programmcode im Hörsaal präsentiert, um zu veranschaulichen, wie die besprochenen KI-Modelle trainiert werden können. In der unteren Folie wird exemplarisch dargestellt, wie das Model durch das Aktualisieren von Rateversuchen lernt.



Links auf der Folie befindet sich der Programmcode mit der hervorgehobener Codezeile, die kritisch für das Verständnis der TN ist. Rechts werden die daraus generierten Ergebnisse visualisiert. Da der Code ausführbar ist, ist der daraus generierte Plot immer kongruent.

Die Lerneinheit verflicht damit die theoretischen Aspekte, veranschaulicht mit Gleichungen und Bilder, mit der praktischen Anwendbarkeit, unterstützt durch den Programmcode. Die Präsentation wird aus ausführbaren Programmcode generiert, welcher vorab den TN zur Verfügung gestellt wird. Die Lerneinheit wird zur weiteren Verwendung aufgezeichnet und mit den TN geteilt.

Weiterführende Literatur und Beispiele

- Knuth, D. E. (1984). Literate programming. The computer journal, 27(2), 97-111.
- Knuth, D. E. (2011). Art of Computer Programming, Volumes 1-4A Boxed Set (3rd. ed.). Addison-Wesley Professional.
- Barba, L. A., Barker, L. J., Blank, D. S., Brown, J., Downey, A. B., George, T., ... & Zingale, M. (2019). Teaching and learning with Jupyter. Recuperado: https://jupyter4edu. github. io/jupyter-edu-book, 1-77.
- Howard, J., & Gugger, S. (2020). Deep Learning for Coders with fastai and PyTorch. O'Reilly Media.
- Waite, J., & Sentance, S. (2021). Teaching programming in schools: A review of approaches and strategies. Raspberry Pi Foundation.

Quellen

Dieser Artikel basiert auf Erfahrungswerten von Lehrenden des Institute of Interactive Systems and Data Science der TU Graz, welche mittels Interviews erhoben wurden. Ein große Danke an David Kerschbaumer, Thorsten Ruprechter und Alexander Steinmaurer für Ihre Zeit und Einblicke.