When and How to Introduce Students to Free and Open-Source Software

Inaugural GVSU Technology Summit

Quinton Quagliano, M.S., C.S.P.

Psychology Department

Table of Contents

1 Introduction

1.1 Disclosures and Disclaimers

- I have no disclosures or conflicts-of-interests related to this presentation
- I am not a software engineer, computer scientist, or other technology-oriented professional by training - but am an enthusiast and hobbyist

1.2 Learning Objectives

- Appreciate *why* we, and our students, should pay attention to how software is published and priced (Sections: Motivation and Purpose)
- Understand the vocabulary used to describe pricing models and source code availability in software (Section: Vocabulary of Software Availability and Pricing)
- Learn about some practical implementation to bring more diverse software to students in class (Section: Examples of Integration)
- Recognize both the advantages and disadvantages of adopting open-source alternatives into instruction (Section: Advantages and Challenges)
- Explore the ways in which exposure to different tools and methods may help produce more technology-literate students (Section: Connection to Tech Literacy)

1.3 Motivation

- The four (hyperbolic) "Evil" Es of software
  - Software is **everywhere** - it's always all around us
  - Software is **essential** - it's a common requirement of navigating the world
  - Software is **elaborate** - but it doesn't look it!
  - Software is **expensive** - and keeps getting more so!
- I want my students able to responsibly navigate these "Evil Es" during and after college, and not feel lost

1.4 Purpose

- Support the liberal arts mission of creating well-(tech)-rounded students
  - Expose students to more alternative tools
  - Help students see the similarities, differences, and quirks of each tool
  - Ensure that when students encounter new software they can adapt easier
- Support projects and software that are free in a time of increasing prices

  - Push back against reliance upon subscription-based and more “locked-down” tools
  - Show students how to build their portfolio and skill set without incurring additional financial burden
  - Make sure that college education does not rely upon the solvency of software businesses and startups

> Important
>
> This is **not** an attempt to insist upon **only** using open-source software

2 Vocabulary of Software Availability and Pricing

2.1 Basic Software Terms

- **End User:** A person who uses a certain piece of software for personal or business use, i.e., You and I
- **License:** some document or text shared with software, written by the creator, that dictates how a certain software, and its source code, can or cannot be used, modified, and shared
  - When used correctly, this is legally binding and ignoring licensee terms can be grounds for lawsuits
  - E.g., MIT License, CC, specific licenses written for proprietary software
- **End User License Agreement (EULA):** An agreement signed by a user of a specific software that recognizes and agrees to the software license terms, and the penalties associated with violating the license
  - I.e., the long set of documents and text we agree to when we sign up for a new service
- **Source code:** the actual code that underlies a certain software or program
  - This is *not* necessarily the files that “ship” with the software
  - This is used to build and modify the program

2.2 Terms of Ownership and Use

- **Non-free:** Software that is priced and/or restricted with limited access to source code. Contrary to the name, it does not necessarily have a cost to use
  - **Proprietary:** A type of non-free software; Something that is owned by an entity via copyright and intellectual property laws, and users are not allowed to make limitless modification to the program or share without authorization
    - E.g., Microsoft 365 (Word, Powerpoint, etc.) - owned by Microsoft, cannot be modified or shared without permission from copyright holder

-
  - **Software as a Service (SaaS):** Proprietary, non-free software priced as a continuous ongoing subscription model, rather than charging a one-time access fee
    - E.g., Adobe Creative Cloud, Anything that says: "Contact our Sales Team"
- **Free and open source (FOSS):** Software that is *intentionally* not priced and able to be adopted, modified, shared, and used without cost - and only some restrictions per the specific license of the software
  - The "open source" part of this, refers to the source code being open and available, i.e., not hidden from the public
  - A subtype of this is **Free, libre, and open source (FLOSS) or copyleft**, which focuses on prohibiting the use of the software to create non-free software
- For the purpose of this presentation, I'll be prioritizing talking about options for using and embracing **FOSS** software

> Important
>
> There's a lot of different ways to describe software price models, so be discerning when looking at options!

3 Examples of Integration

> Important
>
> There are several routes to integration of open-source tools, but all focus on flexibility and agency!

4 Advantages and Challenges

4.1 Benefits

-

4.2 Drawbacks

> **Important**
>
> Introducing open-source tools brings benefits and drawbacks, but the good outweighs the bad!

5 Connection to Tech Literacy

> **Important**
>
> There are several routes to integration of open-source tools, but all focus on flexibility and agency!

6 Conclusion

6.1 Recap

- There are many reasons we should be conscious around what software we use and the growing cost and reliance
- FOSS software *can*, and sometimes *should*, be considered as an alternative when training students for practical work
- *Exposure and experience is key*: we don't necessarily need to persuade students of a certain method of using software, so much as we should help them explore their options

6.2 Parting Message

- Continue using the resources, links, and References provided throughout this presentation to explore the complexity around adopting more open-source software
- If you are an instructor: consider trialing an adoption of open-source tools, like one of my Examples of Integration, in your classrooms
- If you are administrative or support staff, make sure that instructors are aware of alternative options to the most popular tools. IT may even be able to help organize hosting of certain desired tools by faculty

- Encourage students to engage fruitfully with technology not purely as a means-to-an-end, but as a **conscious choice**

6.3 References

Allaire, J., & Dervieux, C. (2025). *Quarto: R interface to quarto markdown publishing system*. https://github.com/quarto-dev/quarto-r

R Core Team. (2025). *R: A language and environment for statistical computing*. R Foundation for Statistical Computing. https://www.R-project.org/

Xie, Y. (2014). Knitr: A comprehensive tool for reproducible research in R. In V. Stodden, F. Leisch, & R. D. Peng (Eds.), *Implementing reproducible computational research*. Chapman; Hall/CRC.

Xie, Y. (2015). *Dynamic documents with R and knitr* (2nd ed.). Chapman; Hall/CRC. https://yihui.org/knitr/

Xie, Y. (2025). *Knitr: A general-purpose package for dynamic report generation in r*. https://yihui.org/knitr/