Nifty Web Apps

Build a Web App for Any Programming Assignment

Kevin Lin, Sumant Guha, Joe Spaniac, Andy Zheng
Paul G. Allen School of Computer Science & Engineering
University of Washington
Seattle, Washington
{kevinl,guhas2,jspaniac,succion}@cs.uw.edu

Figure 1: Example web apps for CS1 and CS2 Nifty Assignments.

ABSTRACT

In this tutorial session, participants will learn how to build interactive web apps to accompany existing console-based programs. Web apps can provide intrinsic motivation as students can interact authentically with their programs and realize their algorithmic ideas over the internet. Whereas other approaches for increasing realism may require significant retooling to fit a particular programming paradigm, we propose an approach that requires few (if any) changes to learning objectives while offering a more engaging medium for students to experience their programming products.

By the end of the tutorial session, participants will be able to write a backend web server that connects to an existing program as well as a frontend that communicates with the server. Participants will learn how to design minimal web app architectures to keep the app accessible and extensible to novices without prior web development experience. Participants will leave with templates for bootstrapping web apps to several CS1 and CS2 Nifty Assignments.

CCS CONCEPTS

• **Social and professional topics → Computing education**.

KEYWORDS

assignments, education, motivation, nifty, tutorial, web apps

Permission to make digital or hard copies of part or all of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for third-party components of this work must be honored. For all other uses, contact the owner/author(s).
SIGCSE '21, March 17–20, 2021, Toronto, Canada
© 2021 Copyright held by the owner/author(s).
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM.
https://doi.org/10.1145/nnnnnnn.nnnnnnn

ACM Reference Format:
Kevin Lin, Sumant Guha, Joe Spaniac, Andy Zheng. 2021. Nifty Web Apps: Build a Web App for Any Programming Assignment. In *52nd ACM Technical Symposium on Computer Science Education (SIGCSE '21).* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 OBJECTIVE

The Nifty Assignments session has consistently attracted computing educators attending the annual ACM Technical Symposium on Computer Science Education (SIGCSE) since 1999 [6]. Over the years, the Nifty Assignments project has gathered over 120 assignments targeted mostly at CS1 and CS2 audiences and spanning a broad range of interests including computer-generated art and music, information representation, interactive simulation, and apps that enable modern day-to-day life [6]. Experience reports and CS education research also indicate that instructors and researchers alike are studying the potential benefits of modernizing the introductory programming experience to improve student engagement and draw in a more diverse discourse in computing [1–3].

This tutorial will help computer programming educators integrate some of these concepts into their existing courses and programming assignments as interactive web apps while minimizing the friction of learning and deploying new technology. Our tutorials will introduce web apps in the context of publicly-distributed Nifty Assignments as well as some of our own in-house assignments demonstrating a variety of ways that web app interactivity can occur, ranging from purely functional input-in, input-out web servers to more complicated dynamics that involve storing, managing, and sharing state between the frontend and backend of the web app. We'll also discuss some of the trade-offs, design decisions, and pedagogical considerations involved integrating web apps into an otherwise traditional CS2 introductory programming course.

2 OUTLINE

5 min	Presenter introductions and web app demos.
15 min	Web app walkthrough, teardown, and architecture talk.
50 min	Small-group tutorials facilitated by presenters.
5 min	Wrap-up, showcase, and contact information exchange.

The authors represent a team instructors and undergraduate teaching assistants who collaboratively developed and refined transitioning text/console-based programming assignments to web apps. The example assignments cover a diverse range of topics and demands for different web app architectures that other instructors are likely to face. Some of these assignments only need to pass text input as a search query and return an object representing the result to the frontend.

Autocomplete Given a text query, automatically provide a suggestion for the rest of the query [8].

Letter Inventory Given a text query, automatically correct any typo or misspelling using cosine similarity.

Search Engine Given a text query, return all matching results from the search engine database [9].

Other assignments need to manipulate and manage state in pieces, which may require coding in frontend web frameworks.

DNA Strand [5, 10].
Language Generator [4].
Election Simulator [7].
Text Classifier

3 EXPECTATIONS

The goal of this tutorial session is to introduce web apps to CS educators primarily teaching introductory programming in imperative languages (such as Python or Java). Concretely, we expect participants to leave the session with the skills and tools to build web apps that communicate text queries between the frontend, backend, and a pre-existing program.

As participants arrive, we will be forming groups of three to six to interact and make connections as prior to the event. A short walkthrough at the beginning to introduce participants to the concept and motivate the active learning that will occur in small-groups. After the demonstration, most of the time will involve participants working in groups to build web apps for their own programming assignment or a Nifty Assignment [6]. The tutorial faciliators will circulate around to different groups and provide one-on-one or small-group instruction as appropriate.

Towards the end of the session, there will be a few minutes for wrapping up, highlighting key points, and collecting feedback via paper or online forms from participants. The success of the tutorial will be determined by several Likert-scale questions about the perceived value of the session as well as free response questions about what we should start doing, stop doing, and continue doing for this tutorial in the future.

It is also important that we create a community of practice to support learning and building engaging technologies for students beyond the 75-minute tutorial session. To this end, we will invite participants to join an online discussion board.

4 VIRTUAL AND HYBRID FORMATS

If the conference will be held virtually or in hybrid format, the most challenging component to adapt will be the small-group tutorial work. We will be particularly explicit about tutorial expectations at the beginning of the event before participants enter small-group work so that everyone is aware of how to support each other within a group, especially if there are not every group is assigned a facilitator for the entire duration of the tutorial session. Facilitators will be particularly cognizant of how long they're spending per group to ensure that every group receives some attention. Participants with web programming experience will be distributed to different groups so that participants can teach and learn from each other.

5 SUITABILITY FOR A SPECIAL SESSION

This special session is structured as a tutorial session. Its format will allow participants to learn about the web app development process that we employed before applying the process to their own programming assignments or Nifty Assignments [6]. The 75-minute duration is the right length for this tutorial since we don't intend to teach all of web programming or even JavaScript in any depth: our web app architecture is intentionally designed to keep things simple for both students and teachers to use, explore, and extend without significant prior web programming experience.

REFERENCES

[1] David J. Malan. 2010. Reinventing CS50. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (Milwaukee, Wisconsin, USA) *(SIGCSE '10)*. Association for Computing Machinery, New York, NY, USA, 152–156. https://doi.org/10.1145/1734263.1734316

[2] Matthew Mcquaigue, Allie Beckman, David Burlinson, Luke Sloop, Alec Goncharow, Erik Saule, Kalpathi Subramanian, and Jamie Payton. 2020. An Engaging CS1 Curriculum Using BRIDGES. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 1317. https://doi.org/10.1145/3328778.3372609

[3] Alvaro E. Monge, Cameron L. Fadjo, Beth A. Quinn, and Lecia J. Barker. 2015. EngageCSEdu: Engaging and Retaining CS1 and CS2 Students. *ACM Inroads* 6, 1 (Feb. 2015), 6–11. https://doi.org/10.1145/2714569

[4] Nick Parlante, Owen Astrachan, Mike Clancy, Richard E. Pattis, Julie Zelenski, and Stuart Reges. 1999. Nifty Assignments Panel. In *The Proceedings of the Thirtieth SIGCSE Technical Symposium on Computer Science Education* (New Orleans, Louisiana, USA) *(SIGCSE '99)*. Association for Computing Machinery, New York, NY, USA, 354–355. https://doi.org/10.1145/299649.299809

[5] Nick Parlante, Thomas P. Murtagh, Mehran Sahami, Owen Astrachan, David Reed, Christopher A. Stone, Brent Heeringa, and Karen Reid. 2009. Nifty Assignments. In *Proceedings of the 40th ACM Technical Symposium on Computer Science Education* (Chattanooga, TN, USA) *(SIGCSE '09)*. Association for Computing Machinery, New York, NY, USA, 483–484. https://doi.org/10.1145/1508865.1509031

[6] Nick Parlante and Julie Zelenski. 2020. *Nifty Assignments.* nifty.stanford.edu

[7] Nick Parlante, Julie Zelenski, John DeNero, Christopher Allsman, Tiffany Perumpail, Rahul Arya, Kavi Gupta, Catherine Cang, Paul Bitutsky, Ryan Moughan, David J. Malan, Brian Yu, Evan M. Peck, Carl Albing, Kevin Wayne, and Keith Schwarz. 2020. Nifty Assignments. In *Proceedings of the 51st ACM Technical Symposium on Computer Science Education* (Portland, OR, USA) *(SIGCSE '20)*. Association for Computing Machinery, New York, NY, USA, 1270–1271. https://doi.org/10.1145/3328778.3372574

[8] Nick Parlante, Julie Zelenski, Baker Franke, Arvind Bhusnurmath, Karen Her, Kristen Gee, Eric Manley, Timothy Urness, Marvin Zhang, Brian Hou, John DeNero, Josh Hug, and Kevin Wayne. 2016. Nifty Assignments. In *Proceedings of the 47th ACM Technical Symposium on Computing Science Education* (Memphis, Tennessee, USA) *(SIGCSE '16)*. Association for Computing Machinery, New York, NY, USA, 588–589. https://doi.org/10.1145/2839509.2844678

[9] Chris Piech and Mehran Sahami. 2020. *Bajillion.* https://web.stanford.edu/class/archive/cs/cs106a/cs106a.1206/assn/15-assignment7.pdf

[10] Keith Schwarz. 2020. *The Adventures of Links.* https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1204/handouts/260%20Assignment%207.pdf