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

```
$ javac Autocomplete.java
$ java Autocomplete
Query: Sea

608660 Seattle, Washington, United States
 33025 Seaside, California, United States
 26909 SeaTac, Washington, United States
 24168 Seal Beach, California, United States
 22858 Searcy, Arkansas, United States

Query:
```

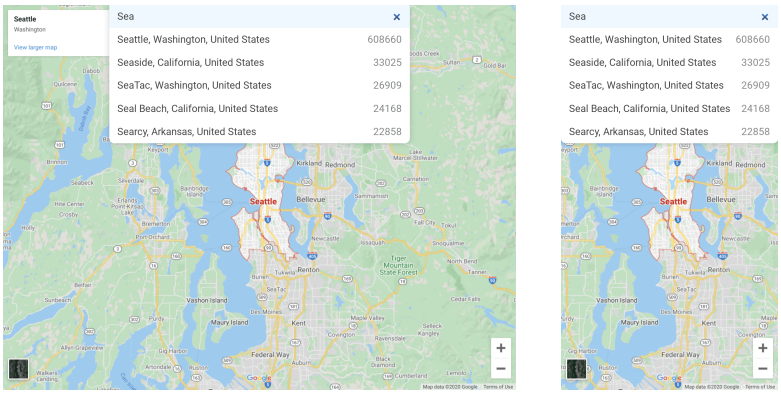

Figure 1: Console input/output compared to a responsive web app for Autocomplete [8].

ABSTRACT

Many students use web apps across a variety of devices to interact with information shared around the world. In this tutorial session, participants will learn to build custom web apps for their programming assignments. Web apps can enable more realistic user interactions than the text-only consoles and desktop GUIs that are still the most common interfaces for Nifty Assignments [6]. Our approach requires 75% less code than similar desktop apps written in Java Swing while presenting an app experience that instructors and students alike can run, modify, and test on demand with their own computers or distribute over the internet with free app hosting. By the end of this tutorial, participants will be able to build simple web apps that execute student-written code to respond to user interaction. The tutorial draws examples from CS1 and CS2 courses offered in Python and Java, but the ideas apply generally.

CCS CONCEPTS

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

KEYWORDS

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

SIGCSE '21, March 17–20, 2021, Toronto, Canada

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 at the annual ACM Technical Symposium on Computer Science Education (SIGCSE) since 1999, gathering over 120 assignments spanning a range of student interests including computer-generated art and music, information representation, interactive simulation, and apps that support daily life [6]. The CS education literature also indicates that there is interest in studying how more realistic programming assignments can increase student motivation and diversify the computing discourse [1–3]. However, adopting new assignments can be time-consuming for instructors.

This tutorial will help participants build interactive web apps as a presentation layer for existing programming assignments. While complex web apps can utilize a variety of web technologies and custom JavaScript to respond to user interaction, our approach instead focuses on simple web apps that take a string query and display a list of results (fig. 1). HTML frameworks such as AMP (https://amp.dev) make building the frontend of a simple web app easier than ever as JavaScript is no longer a prerequisite. The Python and Java standard libraries provide basic HTTP web server implementations for the backend of a simple web app. Compared to a desktop GUI toolkit such as Swing, our example web app for Autocomplete [8] requires 75% less code and includes a new, automatically-updating map.

2 OUTLINE

5 min Presenter introductions and web app demos.
10 min Web app technical overview and template walkthrough.
10 min Groups: Setup, idea brainstorming, design specification.
20 min Groups: Web app backend development.
20 min Groups: Web app frontend development.
5 min Private discussion board showcase and wrap-up.

The presenters developed web apps for 7 week-long programming assignments in a large-enrollment Java CS2 introductory programming course at an R1 university. The instructor (Kevin) will lead the presentation and overview while all of the presenters assist participants during groupwork. Prior to starting this undertaking, the presenters spanned a spectrum of comfort levels with respect to web app development, including some presenters that did not have any web app development experience. Each presenter thus provides unique expertise and perspective on developing web apps to accommodate the variety of participant assignments.

To demonstrate the applicability of building web apps for various assignments, we will draw examples from web apps for 4 Nifty Assignments [4, 5, 7, 8], 2 assignments from other CS1 and CS2 courses [9, 10], and 2 original assignments. While the tutorial session will focus on building simple web apps, we will also discuss the implementation challenges and pedagogical considerations involved in building both simple and complex web apps for our CS2 course. Conversations can continue via a private online discussion board created to sustain the community, provide technical support beyond the session, and inspire new nifty web app ideas.

3 EXPECTATIONS

The goal of this tutorial session is to prepare programming-focused CS educators to build web apps that respond to user interaction by running student-written assignment code. While the approach is not strictly limited to Python or Java programming languages or introductory programming in general, it is easiest to get started in programming languages that provide basic HTTP web server implementations in their standard library. We expect participants to leave the session with the skills and tools to build **simple web apps** that respond to queries by running student code to return a string or a list of strings. Simple web apps can be enhanced by binding results to web services such as Google Maps (fig. 1).

As participants arrive, we will be forming groups of three to six to encourage interaction and build connections prior to the official start. A short walkthrough will occur to provide guided instruction on the necessary web development concepts and frameworks. 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]. A subgoal-labeled script will be provided to each participant to guide their web app development process. Throughout the groupwork, presenters will circulate around and provide one-on-one or small-group support.

During wrap-up towards the end, the instructor will highlight key points and collect participant feedback via paper or online form. It is important that we create a community of practice to support enhancing the student experience beyond the 75-minute tutorial session, so participants will be invited to share their work in a private 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 groupwork. We will be particularly explicit about tutorial expectations at the beginning of the event so that participants know how to support each other within a group and escalate questions, especially if not every group is assigned a presenter for the entire duration. Presenters 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 a structured web app development process and still have time during the session to build a web app for their own programming assignment or a Nifty Assignment. 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 students and instructors alike to use, modify, and extend without 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*. Stanford University. http://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. *Assignment 7: Bajillion*. Stanford University. https://web.stanford.edu/class/archive/cs/cs106a/cs106a.1206/assn/15-assignment7.pdf

[10] Keith Schwarz. 2020. *Assignment 7: The Adventures of Links*. Stanford University. https://web.stanford.edu/class/archive/cs/cs106b/cs106b.1204/handouts/260%20Assignment%207.pdf