# Adaptive IDEs to Support Information Foraging for Novice Programmers

Adam C. Short
University of Tennessee
Knoxville, Tennessee 37996-2250
ashort11@utk.edu

## I. Introduction

There is an abundance of information on the Internet to help programmers and yet, novices may not have the expertise to utilize it. Stack Overflow, documentation for programming languages, man pages, and plenty of other resources exist that aim to help answer the questions of programmers. Stack Overflow itself has over 17 million questions that have been asked. For novice programmers, however, these resources might not be as helpful. For those who are just learning how to program, learning syntax and semantics is hard enough on its own. Foraging through multitudes of resources only adds another layer to master, and with everything else to learn, will only add more difficulties for the novices.

Programmers spend a considerable amount of timing navigating for information, both in and outside their IDE. One study has shown that programmers spend 50% of their time foraging for information [7]. Another study showed that 51% of the participants' navigations ended in some sort of disappointment in the information they received [6]. Not only do programmers spend a considerable amount of time searching for information, whether it be in code or through resources on the Internet, but almost half the navigations aren't helpful. Information foraging is particularly difficult, even more so for novices. This could be due to the fact that the information simply isn't helpful, or it is just hard to make sense of.

Despite all the tools and resources that are available, it seems programming novices still have difficulties using them. How can using the resources that are already available be made easier for these novices? How can novices more easily master foraging for information? In my work, I plan to explore how novices currently use these tools and resources, what advances could be made, and how to apply these changes in a setting apt for novice programers that will aid in foraging for external information.

## II. Background

Much research has been done recently into how IDEs are laid out and their overall design [4]. Code Bubbles [1], for instance, changes the conventional tab-based editor into a canvas-based one where code is show as bubbles. Similarly, Patchworks [3] follows a similar concept except uses 6 "patches" to display code. Both these editors help foraging for information inside of the code itself; however, they do not research foraging for information outside of the IDE. Additionally, two other studies show that (1) a majority of a programmer's time is spent navigating [7], and (2) a programmer's navigations may not lead to the expected information [6]. The primary focus of these studies, however, has been to study the interactions and navigations that happen within the IDE. They still offer key concepts and ideals of how programmers have and want to navigate in their IDE that could tie into my future work.

Additional research has been done into extending the Information Foraging Theory [9]. Notably, one study [8] has explored foraging as a collective group. This study suggests that using some resource populated with information from a group increases the collective discovery. Resources like Stack Overflow use a collective to answer questions, so this should lead to an increase in discovery overall.

Lastly, another paper similarly matches the tool that I have planned to develop. StackInTheFlow [2] suggests Stack Overflow posts based off of errors that develop in runtime of their program; however, this tool ties only into the IntelliJ Java IDEs. This tool had a significant amount of "manual queries", but the "suggested queries" were often ignored. This paper closely models what I want the tool to do for my future research for building a tool for novice programmers to use.

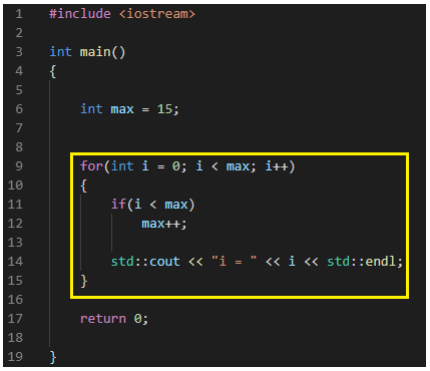
## III. Preliminary Work

My first step was to analyze the size of code and the amount of screen space it requires in an IDE [10]. The size was measured by analyzing the width and length of a typical function for four programming languages. The amount of space required to show the code was calculated for two different screen resolutions, 1920x1080 and 2560x1440. This research has shown that for both resolutions, there is plenty of horizontal screen space available while viewing a single code file on average, regardless of the programming language. Since programmers don't typically view more than one file at a time [1], [5], this space could be utilized in other ways than displaying code.

## IV. Research Plan

*a) Formative Study on Novices Foraging:* Firstly, I will research how current novice programmers, such as students

978-1-7281-0810-0/19/$31.00 



```
#include <iostream>

int main()
{

    int max = 15;


    for(int i = 0; i < max; i++)
    {
        if(i < max)
            max++;

        std::cout << "i = " << i << std::endl;
    }

    return 0;

}
```

Fig. 1. Example Code Segment Identified as Having Issues by the Tool.

in early level computer science courses, search and use information across the Internet. I will do this by running a user study with participants from my university's introduction to computer science class and other beginner courses. I will then ask them to complete a series of trivial tasks that will require the participants to use outside resources, such as the Internet. I then shall record multiple factors such as what resource the participant uses, how the participant uses the information they find, whether or not the information was relevant, and how long it takes to find useful information. The actions will be recorded on the computer itself, and a person will also be present to record any additional notes that are relevant.

*b) Tool Design & Development:* I will then develop a tool that will aim to help novices find and make use of meaningful information through suggestions without leaving their IDE. This tool will make use of the available screen space that most programmers do not utilize [1], [10]. It will pull information from commonly used resources, such as Stack Overflow and any other commonly accessed resources from the first user study, and present the information to the programmer. The information will be pulled in one of two ways: the first being that the programmer requests it, and the second by suggesting information that would be useful to the programmer pertaining to certain pieces of code. An example of information being suggested on a piece of code that is troubling the programmer is shown in Fig. 1 If the programmer hovers a section surrounded by the yellow box, it will give them an option to show a Stack Overflow post, or a similar resource, that relates to their problem in the IDE adjacent to their code.

*c) Evaluation:* I will lastly run a case study where I let novice programmers use the tool that I have built. The novice programmers will consist of a similar grouping of students as the first study from an introductory computer science course. I shall collect data on the amount of information presented, whether the information was considered, and the usefulness of the information. This information shall be recorded similarly to the first study.

*d) Comparison of Results:* For the final step, I will analyze the data that I have collected from both studies. I will compare the results from the first and second study to determine whether or not any improvements where made. Furthermore, depending on the success of this study, I would like to release this tool in a "real-world" setting to determine its effectiveness across different environments.

## V. Conclusion

I have summarized some of my initial research and work into providing an easier method for novices to forage for information across the Internet and laid forth my plan of how to proceed. As I continue this work, my major concern is not only if the tool itself is useful, but the information provided is useful. Will this tool actually help novice programmers by offering useful information, or will it be a distraction? Will the tool help educate, or will it detract and become a clutch that novices rely on? I will continue to research these questions, as well as building a tool that allows me assess the results for these questions and allow easier foraging of information for novices.

## References


[1] A. Bragdon, R. Zeleznik, S. P. Reiss, S. Karumuri, W. Cheung, J. Kaplan, C. Coleman, F. Adeputra, and J. J. LaViola, Jr., "Code bubbles: A working set-based interface for code understanding and maintenance," in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, ser. CHI '10, 2010, pp. 2503–2512.

[2] C. Greco, T. Haden, and K. Damevski, "Stackintheflow: Behavior-driven recommendation system for stack overflow posts," in *Proceedings of the 40th International Conference on Software Engineering: Companion Proceeedings*, ser. ICSE '18. New York, NY, USA: ACM, 2018, pp. 5–8. [Online]. Available: http://doi.acm.org/10.1145/3183440.3183477

[3] A. Z. Henley and S. D. Fleming, "The patchworks code editor: Toward faster navigation with less code arranging and fewer navigation mistakes," in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, ser. CHI '14, 2014, pp. 2511–2520.

[4] A. Z. Henley, S. D. Fleming, and M. V. Luong, "Toward principles for the design of navigation affordances in code editors: An empirical investigation," in *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, ser. CHI '17. New York, NY, USA: ACM, 2017, pp. 5690–5702.

[5] A. J. Ko, B. A. Myers, M. J. Coblenz, and H. H. Aung, "An exploratory study of how developers seek, relate, and collect relevant information during software maintenance tasks," *IEEE Trans. Softw. Eng.*, vol. 32, no. 12, pp. 971–987, Dec. 2006.

[6] D. Piorkowski, A. Z. Henley, T. Nabi, S. D. Fleming, C. Scaffidi, and M. Burnett, "Foraging and navigations, fundamentally: Developers' predictions of value and cost," in *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*, ser. FSE 2016. New York, NY, USA: ACM, 2016, pp. 97–108. [Online]. Available: http://doi.acm.org/10.1145/2950290.2950302

[7] D. J. Piorkowski, S. D. Fleming, I. Kwan, M. M. Burnett, C. Scaffidi, R. K. Bellamy, and J. Jordahl, "The whats and hows of programmers' foraging diets," in *Proc. SIGCHI Conf. Human Factors in Computing Systems*, ser. CHI '13, 2013, pp. 3063–3072.

[8] P. Pirolli, "An elementary social information foraging model," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '09. New York, NY, USA: ACM, 2009, pp. 605–614. [Online]. Available: http://doi.acm.org/10.1145/1518701.1518795

[9] P. Pirolli and S. Card, "Information foraging in information access environments," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '95. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 1995, pp. 51–58. [Online]. Available: http://dx.doi.org/10.1145/223904.223911

[10] A. C. Short and A. Z. Henley, "Towards an Empirically-Based IDE: An Analysis of Code Size and Screen Space," in *2019 IEEE Symp. Visual Languages and Human-Centric Computing (VL/HCC '19)*, October 2019.