Software Engineering User Study Recruitment on Prolific: An Experience Report

Brittany Reid, Markus Wagner
The University of Adelaide
Australia

Marcelo d'Amorim
Universidade Federal de Pernambuco
Brazil

Christoph Treude
The University of Melbourne
Australia

ABSTRACT

Online participant recruitment platforms such as Prolific have been gaining popularity in research, as they enable researchers to easily access large pools of participants. However, participant quality can be an issue; participants may give incorrect information to gain access to more studies, adding unwanted noise to results. This paper details our experience recruiting participants from Prolific for a user study requiring programming skills in Node.js, with the aim of helping other researchers conduct similar studies. We explore a method of recruiting programmer participants using prescreening validation, attention checks and a series of programming knowledge questions. We received 680 responses, and determined that 55 met the criteria to be invited to our user study. We ultimately conducted user study sessions via video calls with 10 participants. We conclude this paper with a series of recommendations for researchers.

1 INTRODUCTION

Prolific is an online recruitment platform with over 150,000 active participants as of December 2021[1]. Members are encouraged to answer demographic questions, which can then be used by researchers to narrow the pool of eligible participants for their studies. However, the existing prescreening features are insufficient; questions are limited and rely on self-reporting. This self-reporting can be an issue, as Software Engineering user studies need to recruit *programmers*, but non-programmer participants often identify as programmers to be admitted into more studies.

We recruited 10 participants for a user study involving the completion of simple programming tasks in Node.js using a code search tool (NCQ) [2]. While existing work has evaluated the accuracy of self-reported programming skill, establishing its lack of reliability [3] [1], this study documents our experiences recruiting participants for **user studies** on Prolific. The negative effect of non-programmer participants can be significant as user studies often require researcher supervision.

While a generic prescreening option for programming skills exists, to recruit programmers of a specific language, Prolific recommends[2] first running a custom screening survey to identify the desired demographic, then inviting them to a second study. To recruit Node.js programmers we surveyed 680 participants, using a set of programming knowledge questions derived from Danilova et al. [1]. We also used attention checks and validated the answers to prescreening questions and found that 33% of participants who completed our survey answered inconsistently, providing different answers in our survey than in their prescreening questions.

We identified 55 Node.js programmers from a pool of 206 self-reported Node.js programmers, for a total cost of £193. Our findings suggest, for example, that researchers should not solely rely on self-reported programming skill.

2 METHOD

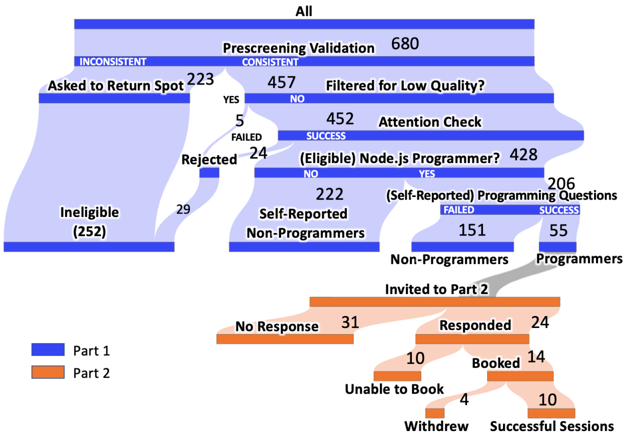

Figure 1: Diagram of participant recruitment process.

We designed a two-part study consisting of a *screening survey* and *video call* study. All participants who complete Part 1 were paid £0.35, and £7.50 for Part 2. We ran Part 1 in multiple waves with different study sizes, first to pilot the study and then until we reached 10 successful user study sessions. Figure 1 illustrates the different steps of our participant recruitment process.

All studies on Prolific require a description and link to a survey. Researchers cannot collect any personal details, instead they must collect participant's Prolific IDs. A completion code must be given at the end. We used Google Forms to host our surveys; the full surveys are available on GitHub.[3]

2.1 Screening Survey (Part 1 of 2)

Part 1 was restricted to participants with an approval rating above 95 and who answered yes to the "programming skills", "video call" and "English fluency" prescreeners. In our description, we included

[1] https://www.prolific.co/

[2] https://researcher-help.prolific.co/hc/en-gb/articles/360015732013

[3] https://github.com/Brittany-Reid/SE-Prolific-User-Study-Recruitment

ICSE 2022, May 21–29, 2022, Pittsburgh, PA, USA

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/1122445.1122456

the fact this was a two-part study, the pay rate, and a list of requirements including programming skill and ability to execute Node.js code, as participants are not shown what prescreeners are enabled on a study.

The survey had four sections: prescreening validation (P), demographics (D), programming knowledge (K) and the attention check (A). Providing inconsistent prescreening answers took participants to a page of the Google Form asking them to return their place in the study (freeing it for another participant). Those who answered "Never" to programming in Node.js (D3) were filtered early, and taken to a section of the Google Form asking them to submit.

Table 1: Example of programming knowledge questions (K)

K5	Look at the above code, what is the parameter of the function?
K6	What is the output of the above code snippet?
K7	Execute the above code in Node.js, what is the result?

Table 1 shows an example of our programming knowledge questions, adapted from Danilova et al. [1]. We selected the six most effective questions (K1-K6). As participants would need to use Node.js in our user study, we also added an additional question (K7), asking participants to demonstrate their ability to execute and provide the output of some Node.js code. We devised a code snippet that could not easily be guessed, but was also easy to read and clearly not malicious (no external libraries or file system access).

To ensure that participants read instructions carefully, an attention check (A1) was also included. Attention check questions ask participants to answer based on some proceeding instructions; in our case, we first told participants to answer the next question with 'Java', then asked them 'Based on the above text, what is your favourite programming language?'. Attention check failures were rejected.

2.2 Video Call Booking (Part 2 of 2)

Participants who answered all programming questions correctly were invited to Part 2 on Prolific. The survey link was used to ask for consent to being recorded, and provide a link to book sessions through YouCanBook.me. Details about the required programming skills and an example programming task were also provided. On booking a session, participants were provided a completion code. We contacted participants who booked a session with our Skype details via Prolific, also reminding them of Prolific's policies on collecting personal information and that they should use their Prolific ID as their username.

3 RESULTS

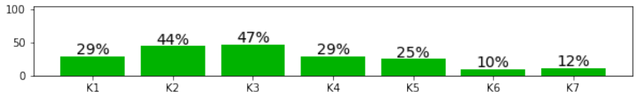

Figure 2: Percentage of correct answers (non-programmers)

We received 680 responses to our screening survey, however, there were an additional 664 participants who attempted our study and did not submit before returning their spot or timing out. Participants who returned their spot or timed out did not receive any payment. Figure 1 shows the breakdown of participants at each stage of our recruitment process. We found that 33% of participants gave inconsistent responses to prescreening validation (P1 and P2), justifying the importance of these questions. Another 24 failed attention checks and 5 were filtered for insufficient effort (nonsensical or random responses), leaving us with 428 eligible participants. 206 participants self-reported as Node.js programmers, and only 55 passed all questions. While only 24 of these participants responded to the invitation to Part 2, this may not reflect total interest; once all places filled, no more participants could respond. 10 participants returned their place without booking, while 14 booked a session. Of these 14, 10 video calls were successful, while 4 withdrew during the session. Reasons for withdrawal included lack of confidence in programming ability and anxiety programming under observation.

Of the programmer subset, 47% reported working in a software related industry, compared to just 18% in the eligible set, indicating that the programming knowledge questions did indeed identify programmers. 85% of programmers were male, in line with other studies of software engineers [3]. Figure 2 shows the percentage of non-programmers who got each question correct; the most effective questions were K6, K7 and K5.

4 RECOMMENDATIONS

The results of our screening survey and user study leave us with a set of recommendations.

- Researchers should not rely on self-reported programming skill when conducting studies. We identified that only 12% of eligible participants, who self-reported having programming skills using Prolific's prescreening questions, and only 27% of participants who self-reported as programming in Node.js, met our criteria. We recommend that researchers use other measures such as the programming questions described in our paper.
- Prescreening validation is important; 33% of participants gave inconsistent responses.
- Researchers should run a pilot study with a small number of participants to test their study design and gather data on the number of non-programmers they will need to account for.
- Be clear about the requirements of each study, such as the required programming skill. Participants cannot see enabled prescreeners, so we included the need for programming skills in the description for Part 1. In an effort to reduce withdrawal rate, we also included an example task when booking Part 2, so that participants would be confident in meeting our requirements.

ACKNOWLEDGMENTS

This research is supported by an Australian Government Research Training Program (RTP) Scholarship.

REFERENCES

[1] Anastasia Danilova, Alena Naiakshina, Stefan Horstmann, and Matthew Smith. 2021. Do you Really Code? Designing and Evaluating Screening Questions for Online Surveys with Programmers. In *ICSE*. 537–548.

[2] Brittany Reid, Keila Barbosa, Marcelo d'Amorim, Markus Wagner, et al. 2021. NCQ: code reuse support for Node. js developers. *arXiv preprint arXiv:2101.00756* (2021).

[3] Mohammad Tahaei and Kami Vaniea. 2022. Recruiting Participants With Programming Skills: A Comparison of Four Crowdsourcing Platforms and a CS Student Mailing List. In *CHI*. 1—-16.