ActiveState®

Developer Survey 2019

Open Source Runtime Pains

Table of Contents

Introduction

ActiveState ran its annual developer survey again this year to better understand open source runtime pains.

We sought responses from developers around the globe and garnered a total of 1,250 responses from 88 countries.

The survey results are compared against 2018 baseline metrics to understand the evolving challenges faced by coders (developers, engineers, data scientists, Q&A, etc.) when working with open source runtimes. This year we also added a few more questions to better understand some of the insights obtained in our 2018 Developer Survey, Open Source Runtimes.

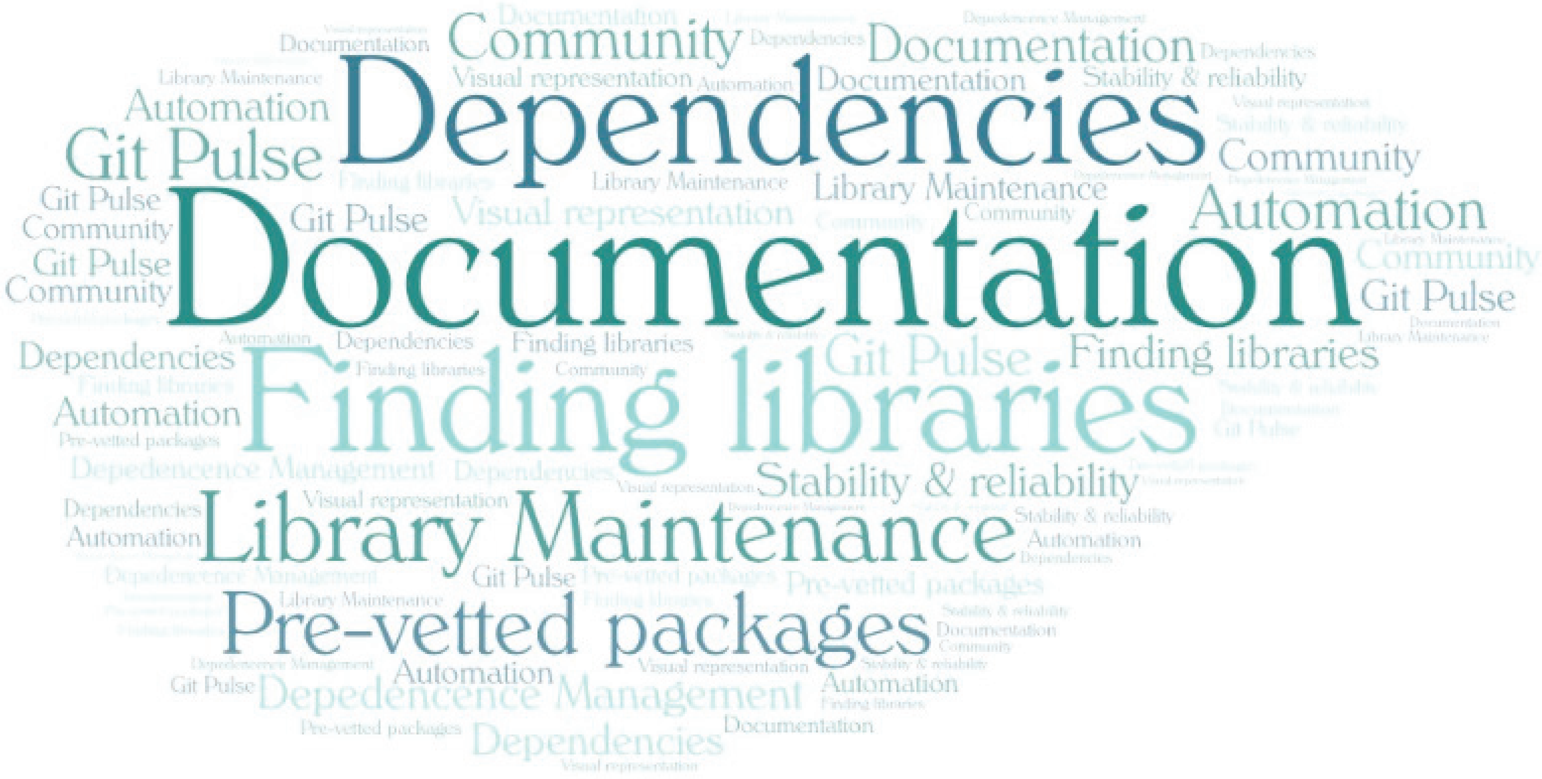
The resulting survey data is an invaluable tool to measure and track progress towards solving open source runtime pains that developers are experiencing.

ActiveState has been working in open source for over 20 years and we are passionate about making open source easy for enterprises and inspiring coder passion with elegant solutions.

What follows are the results of ActiveState's 2019 Developer Survey, Open Source Runtime Pains.

Make it easier to work with open source languages

Here are a few things respondents told us would make their life easier when working with open source projects. It highlights a subset of requirements and pain points within specific themes.



Better system for figuring out which of the many modules / packages / projects in the OSS world are the ones I want to use.”

Proper documentation with multiple examples from industry.”

Ability to download all of the libraries being pre-vetted.”

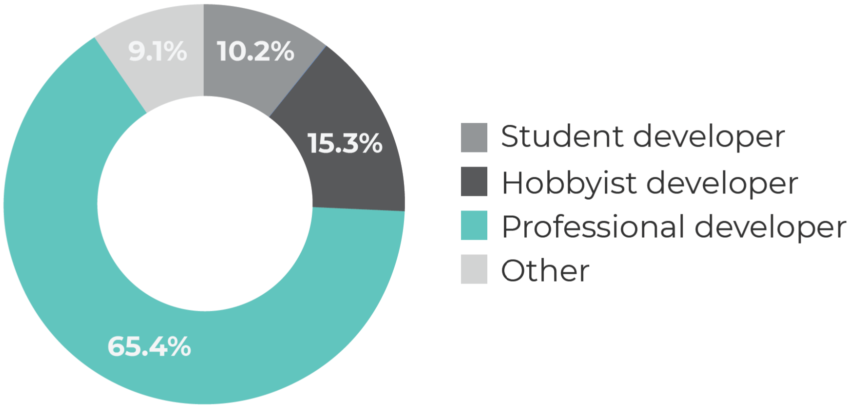
Finding and evaluating open source packages for specific needs - stability, extent of support, modest tutorials or examples.”

Integrated support for a multi-programming-language environment.”

More visual representation of dependencies; better AI-based relationship exposure.”

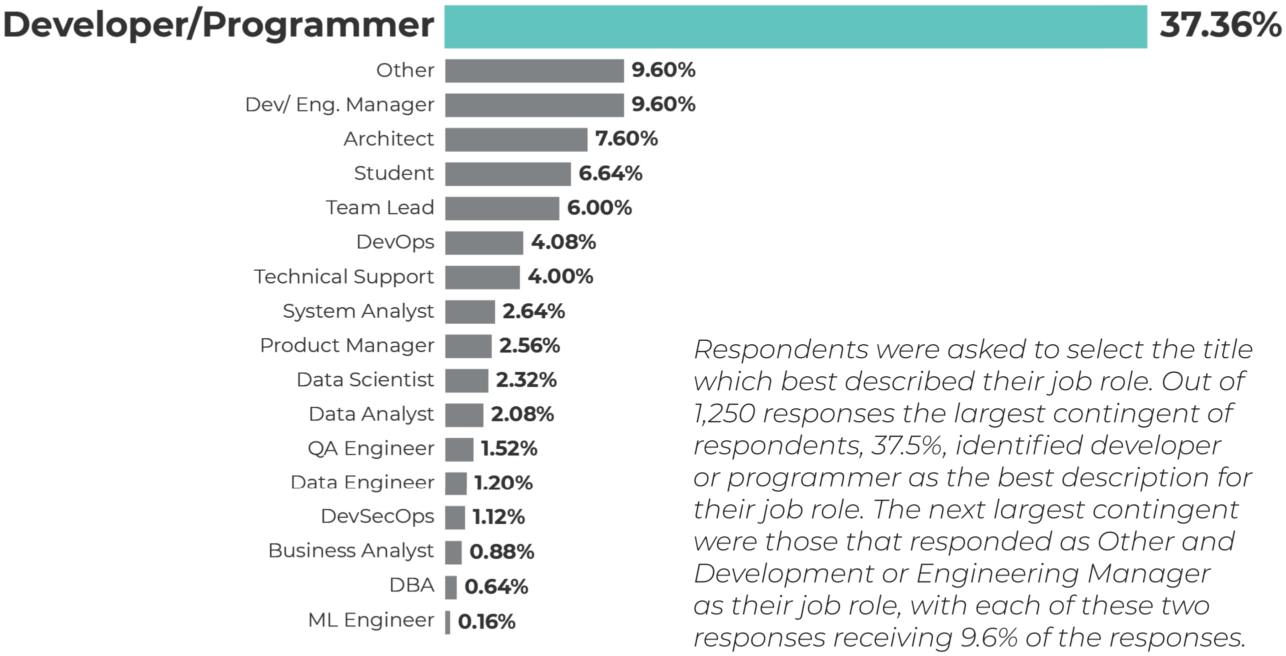
Job Function Questions

Primary Role



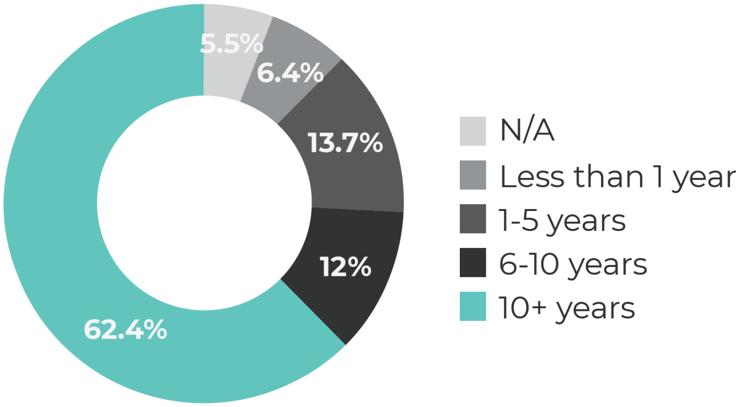
This year we asked respondents to select their primary role. Out of 1,250 responses the largest contingent of respondents, 65.4%, selected professional developer. The remainder of respondents were hobbyist developer (15.3%), student developer (10.2%) or other (9.1%).

Job Role Description



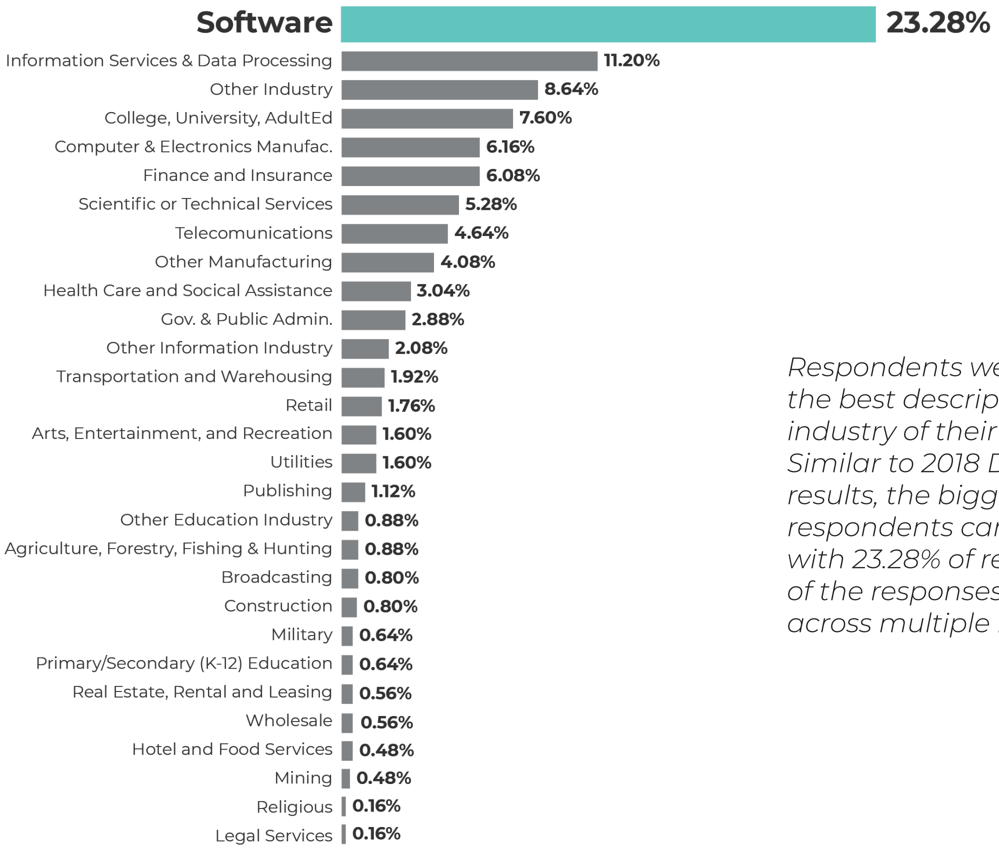
Respondents were asked to select the title which best described their job role. Out of 1,250 responses the largest contingent of respondents, 37.5%, identified developer or programmer as the best description for their job role. The next largest contingent were those that responded as Other and Development or Engineering Manager as their job role, with each of these two responses receiving 9.6% of the responses.

Years Working in a Technical Role



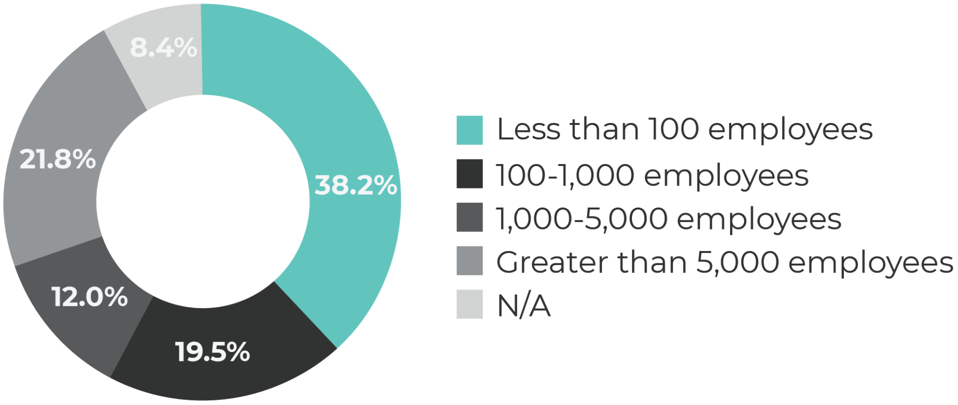
Respondents were asked how long they'd been working in a technical role. The vast majority of survey respondents, 74.4%, have spent have spent 6 or more years in a technical role, with the most significant portion of total respondents, 62.4%, having spent 10 or more years in a technical role. These results closely mirror those from 2018, with 77% and 63% respectively.

Organization's Principal Industry



Respondents were asked to select the best description for the principal industry of their organization. Similar to 2018 Developer Survey results, the biggest contingent of respondents came from software, with 23.28% of respondents. The rest of the responses were staggered across multiple industries.

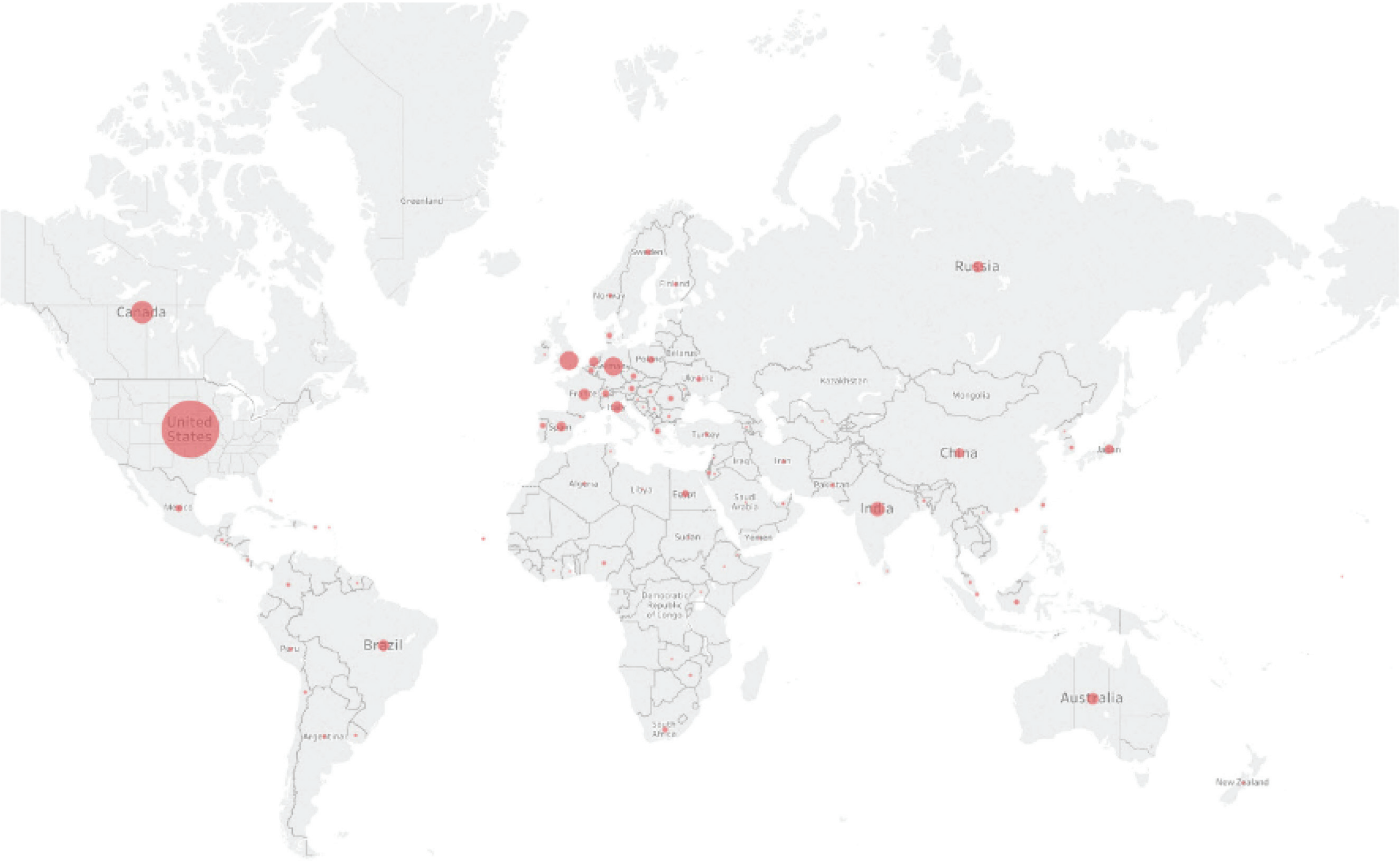
Organization Size by Full-time Employees



Respondents were asked to indicate the size of their organization by the number of full-time employees. A little over half of respondents, 53.3%, were spread across organizations of 100+ employees. And 38.2% were in organizations of less than 100 employees.

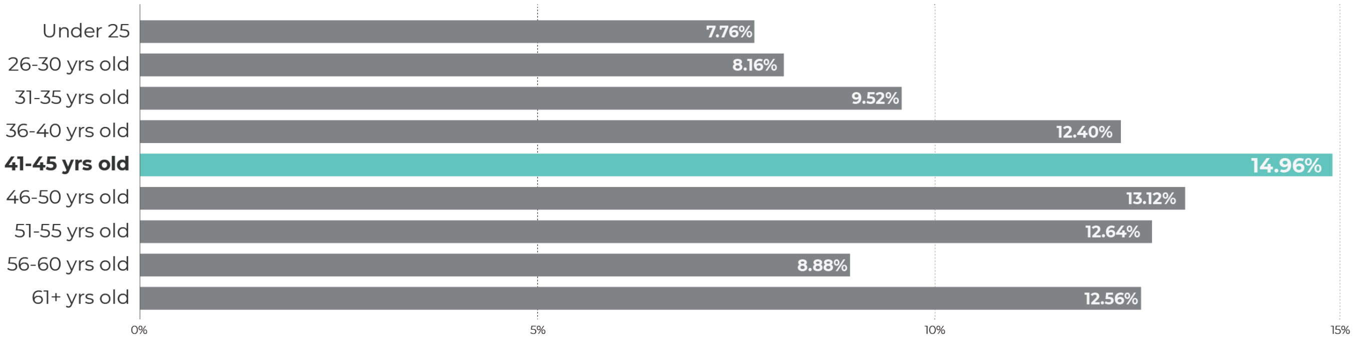
Demographic Questions

Country of Residence



Respondents were asked in which country they lived. The 1,250 individuals spanned 88 countries (versus 92 from 2018 Survey results). Close to half of all respondents, 47.36%, reside in the United States. The next most recurring responses being Canada and Germany, with 7.28% and 5.12% naming them as their country of residence, respectively.

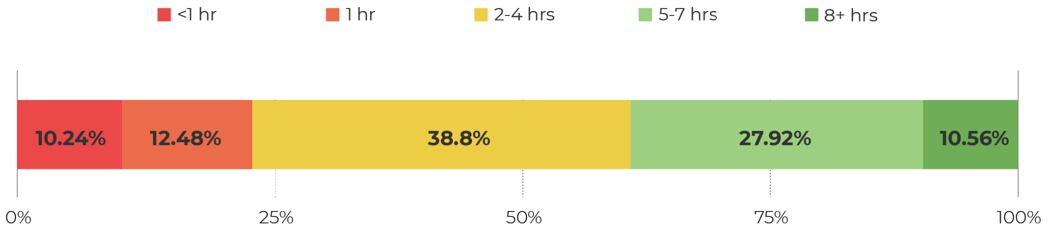
Age Range



Respondents were asked to select their age range. The greatest contingent of respondents, 14.96%, were 41-45 years old. 37.84% of respondents were 40 years old or younger and 47.2% of respondents were 46 years old or older.

Productivity Questions

Programming Hours

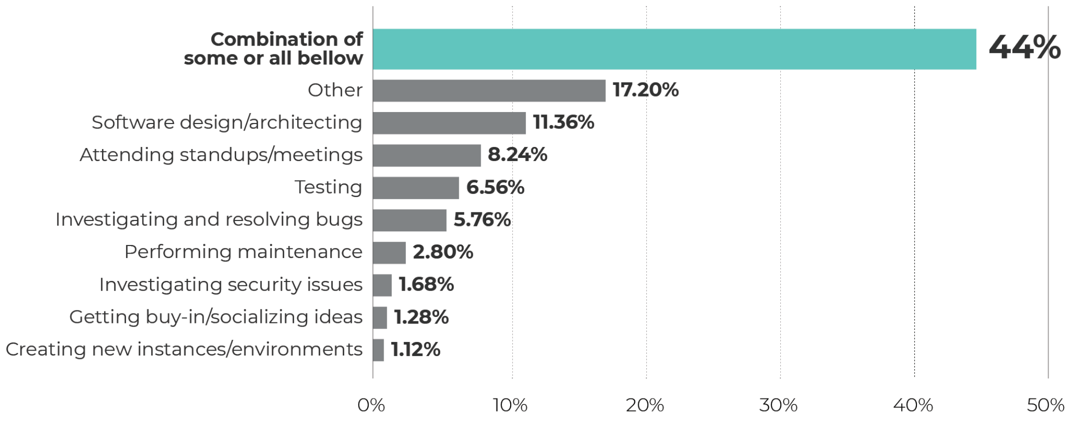


Respondents were asked how many hours they spent programming, on a typical day. Out of 1,250 responses, the biggest portion of surveyed respondents, 38.8%, spend only 2-4 hours a day programming. This is similar to 2018 results which found 37% of respondents spend 2-4 hours a day programming.

In contrast only 27.92% of respondents spent 5-7 hrs a day programming unlike 2018 results which found 31% of respondents spent 5-7hrs a day programming.

Most notably, ActiveState's 2019 Developer Survey results illustrate that as many as 61.52% of respondents now spend 4hrs or less programming, while in 2018 only 51% of respondents spent 4hrs or less programming. This is an approximate 20% decrease in time spent programming. And only 10.56% of respondents spend 8 or more hours programming in contrast to 2018's 19%, a decrease of almost half.

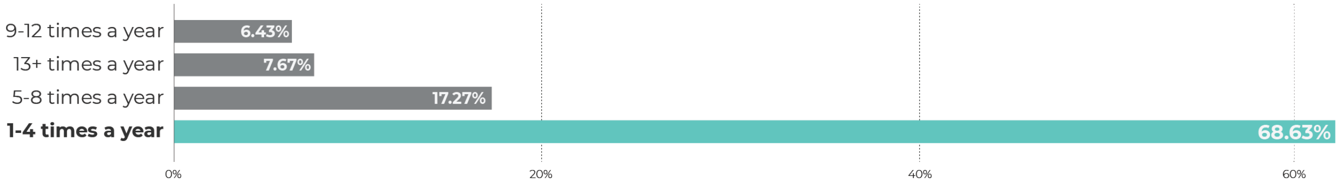
Bulk of Non-Programming Time



In our 2018 Developer Survey results we identified how much time wasn't spent programming. This year we wanted to better understand where time was spent when not programming.

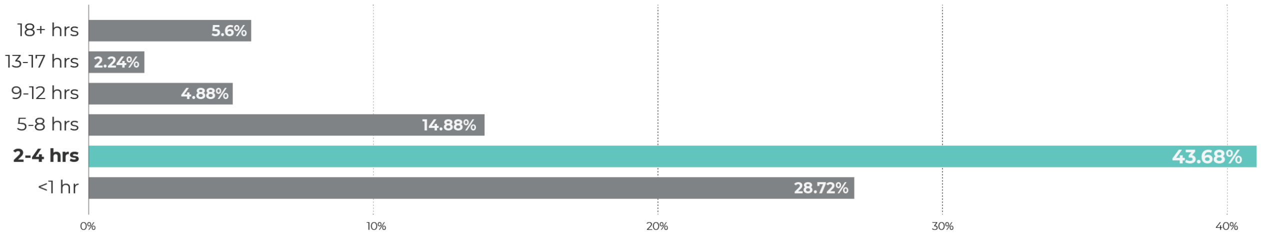
We asked respondents, when not programming, where was the bulk of their time spent? The largest contingent of responses, 44%, had to dedicate time to a multitude of activities including meetings, testing, maintenance and even socializing ideas. However, the biggest single activities taking time from developers programming was software design/architecting, 11.36% respondents, followed by attending standups/ meetings, 8.24% of respondents.

Frequency of Dev Environment Set-up



This year we asked respondents to tell us the frequency with which they setup a fresh dev environment. Almost 69% of respondents create a new dev environment between once a year and once a quarter, but a little more than 31% build between 1 and 2 new dev environments each quarter. Notable is that nearly an equal amount of respondents, 6.43%, setup dev environments 9-12 times a year, as those that do 13+ times or more each year, 7.67% of respondents.

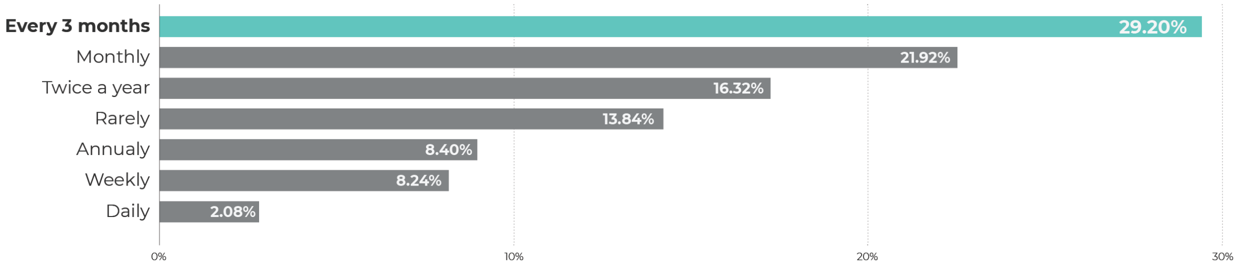
Time for Dev Environment Setup



In order to understand the impact on productivity that setting up a fresh dev environment would have, we wanted to understand how long, on average, it took respondents to setup one up. The greatest contingent of respondents, 43.68% took 2-4 hours. Although 28.72% took less than an hour, almost an equal amount, 27.6%, took 5 or more hours.

From the question above, if you're only setting up a fresh dev environment in less than an hour once a year, the impact to productivity is negligible. Contrast this with someone setting up a dev environment more than monthly, eg 13 times per year, and taking 18 hours to do it. That's a total of 234 hours out of a work year of 2,000 hours. So the single task of setting up a dev environment could be as much as 10% of total dev time.

Frequency of New Software Projects



We asked respondents how often they started a new software project. In aggregate, 61.4% of respondents start a new project at least once a quarter (every three months). This result is almost identical to 2018's 61%. However, 2019 results show a drop in the respondents that start projects once a week, from 12% to 8.24%, and an increase in the respondents that start projects once a quarter, from 26% to 29.2%. While ~32% of respondents start new projects on a monthly or more frequent cadence, the survey shows that only ~14% build a new development environment for that project, contrary to best practices.

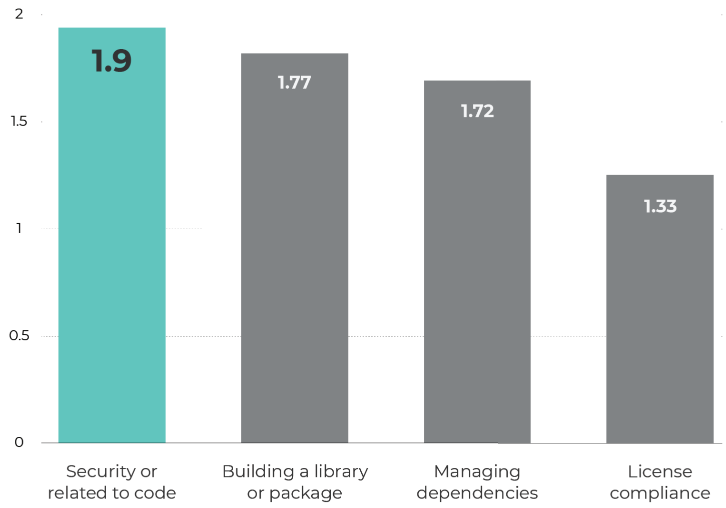
Time Spent on Issues (Weighted & Per Issue)

Last year we asked respondents, over a typical week, how much of their time was spent managing dependencies or dealing with dev tools. (Note: 73% of respondents spent part or most of their time managing dependencies or dev tools, and only 25% of respondents rarely spent time managing dependencies or dev tools.) This year we wanted to better understand how much relative time, over a typical week, was spent dealing with the following issues:

- Security requirements
- Building a library or package
- Managing dependencies
- License compliance or open source licenses

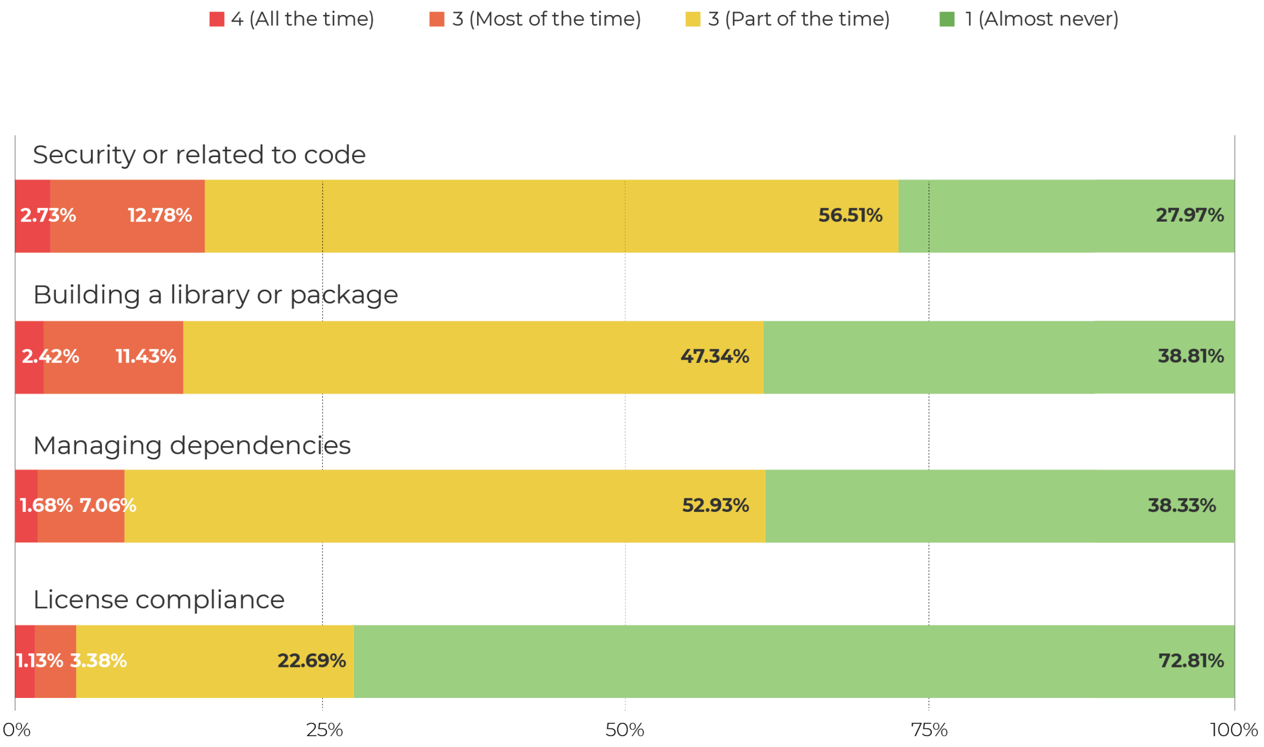
Below are the weighted results as well as results for each of the four issues.

Weighted Results



The weighted results show that the greatest amount of time was spent on issues related to security or code. Time spent on issues related to building a library or package versus managing dependencies was almost equal, 1.77 vs 1.72 respectively. Respondents spent only 70% of time they dedicated to security addressing license compliance issues.

Specific Results

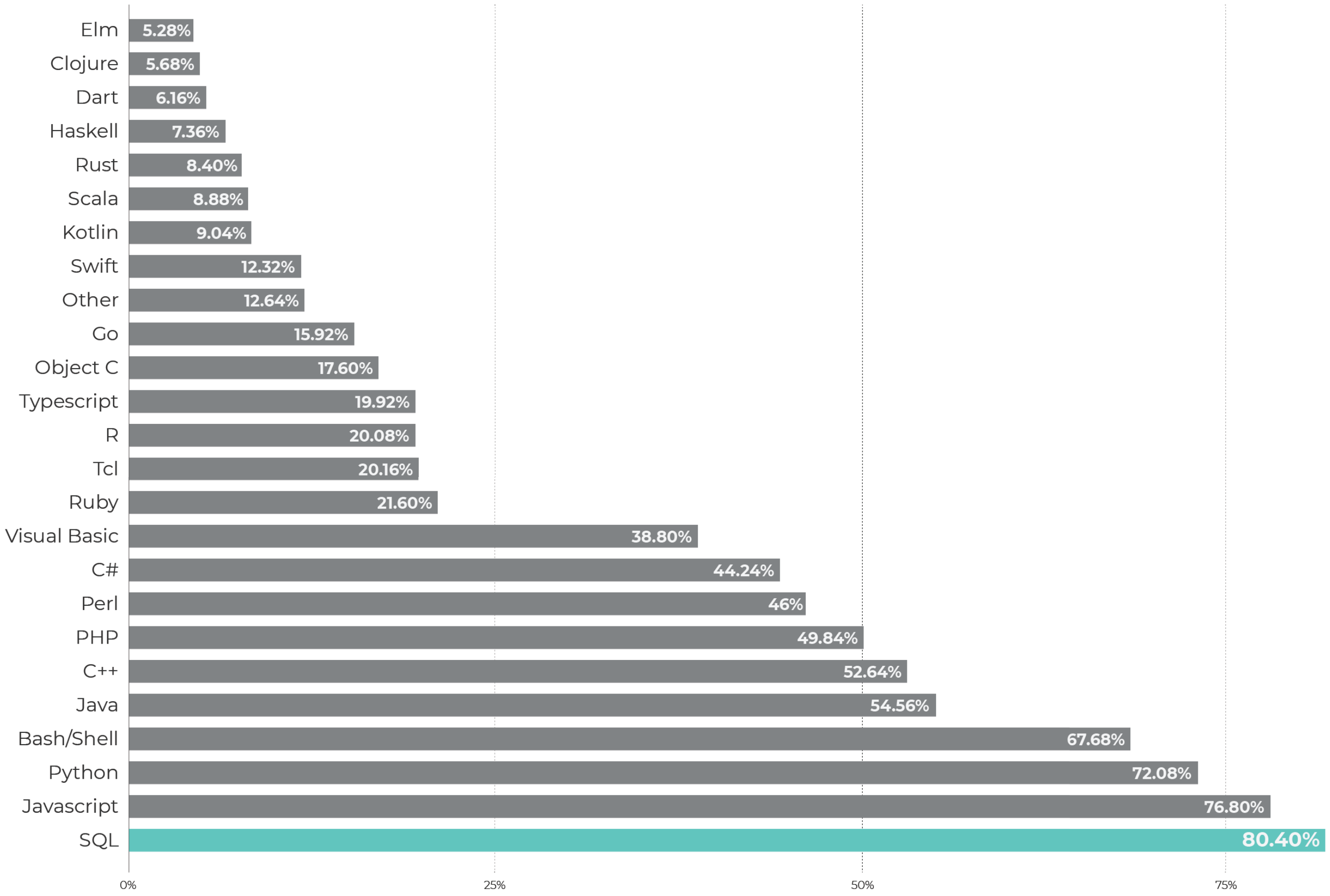


Stacked up against each other the time spent on each of the issues becomes much more significant. For example, 72.81% of respondents stated they almost never spent time on license compliance or open source licenses. Conversely, 61.67% of respondents spend part to all of their time managing dependencies. It should also be noted that this is a significant decrease over last year, when 75% of respondents said they spent part to all of their time managing dependencies and development tools. The delta between 2018 and 2019 results could be due to considering the total combined time spent on each managing dependencies and development tools.

Lastly, ActiveState's 2018 Developer Survey asked respondents the frequency of encountering an issue when building a library or package. In 2018, 71% of respondents encountered issues part of the time to all of the time when building a library or package. And this year, 61.19% of respondents responded that they spend part to all of their time building a library or package.

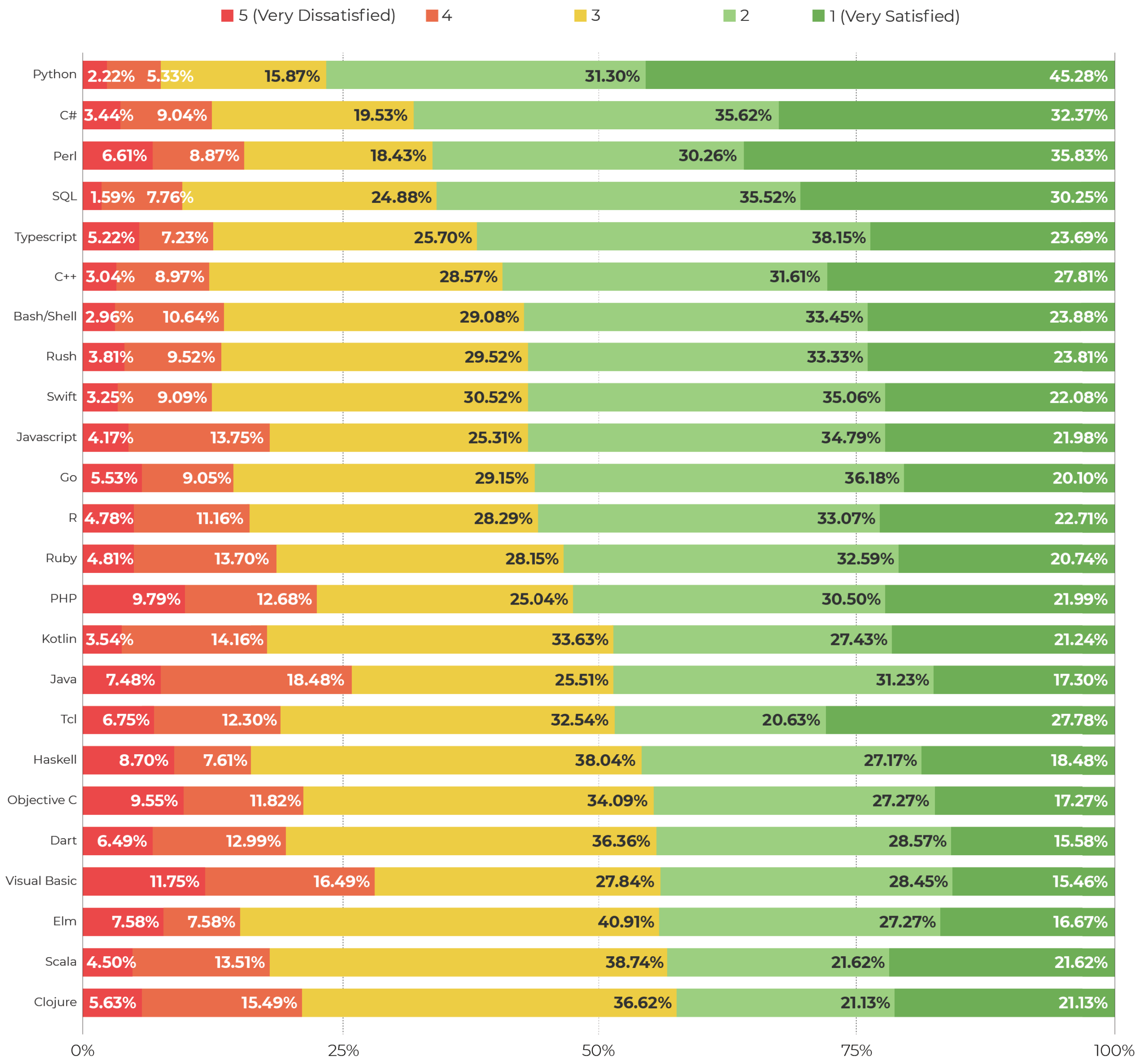
Language Questions

Language Popularity



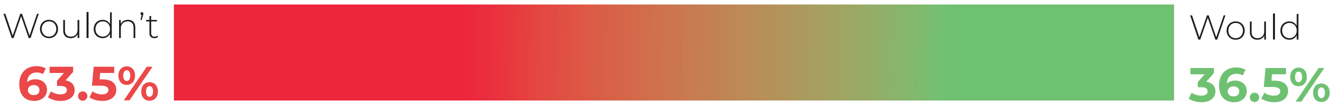
When respondents were asked which programming languages they used, the top three most popular languages were SQL (80.4%), Javascript (76.8%) and Python (72.08%).

Satisfaction



Python was the third most popular language, and also had the highest satisfaction level, with 76.58% of respondents giving Python a satisfaction level of somewhat satisfied or greater. This is a marked increase over 2018 results when only 68% of respondents were somewhat satisfied or greater with Python. The most popular language, SQL, rated with a 66.09% somewhat satisfied or greater satisfaction level. Conversely, Clojure had the lowest satisfaction level, with only 42.26% of respondents rating it with a somewhat or greater satisfaction level. And the least popular language, Elm, had only 43.94% of respondents who had a satisfaction level of very satisfied or greater.

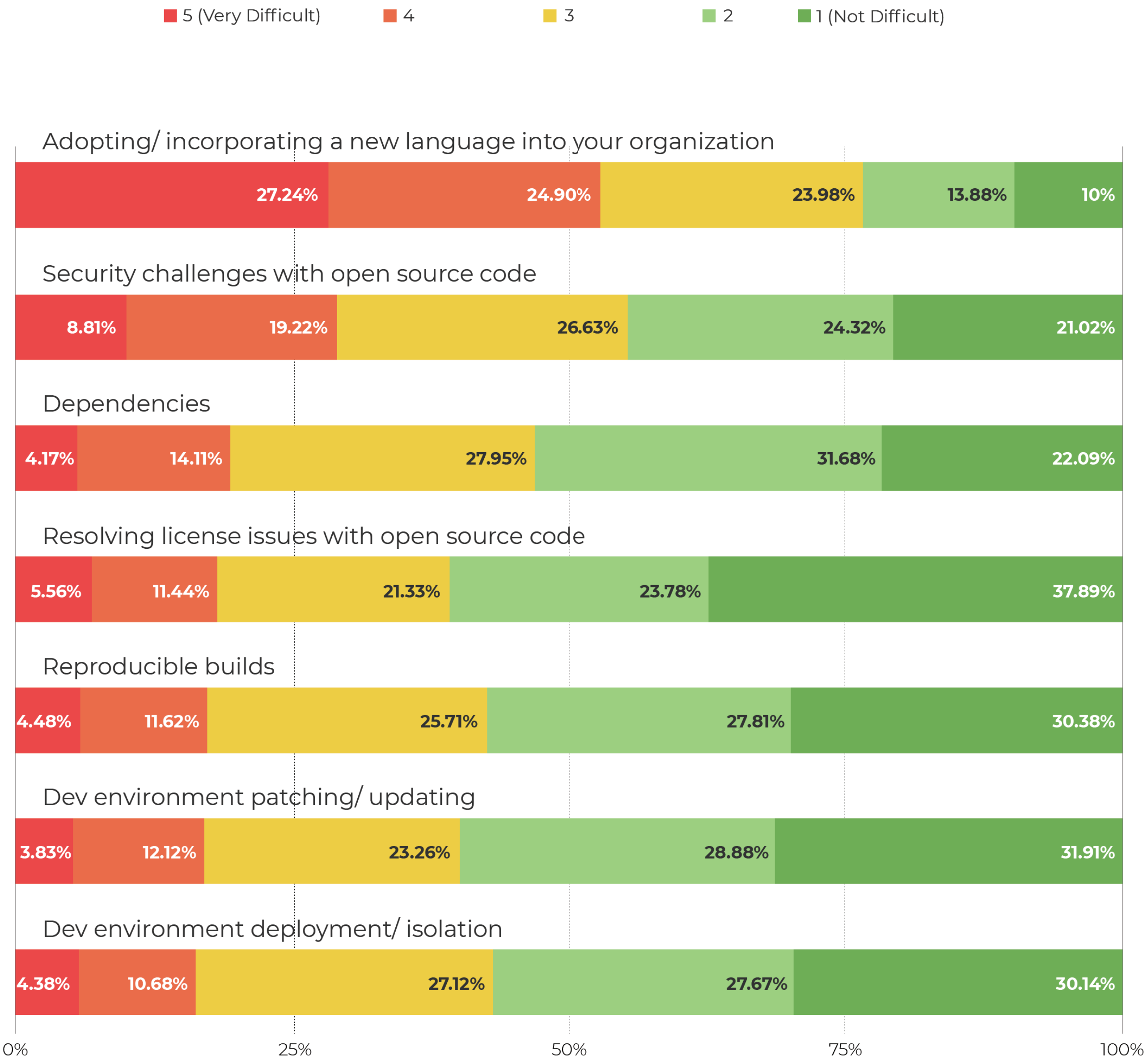
Adding a Language



We asked respondents if they’d ever decided not to use the best tool for the job because it meant adding a new programming language. In other words, the pain of adding the new language would outweigh the benefits of using the tool in question. 63.5% of respondents wouldn’t add a language when the pain outweighed its benefits, and 36.5% would add the language. This is similar to 2018 results, in which 67% of respondents would choose not to add a new programming language when the pain of adding the language would outweigh the benefits of said language.

Setup, Deployment, Runtime

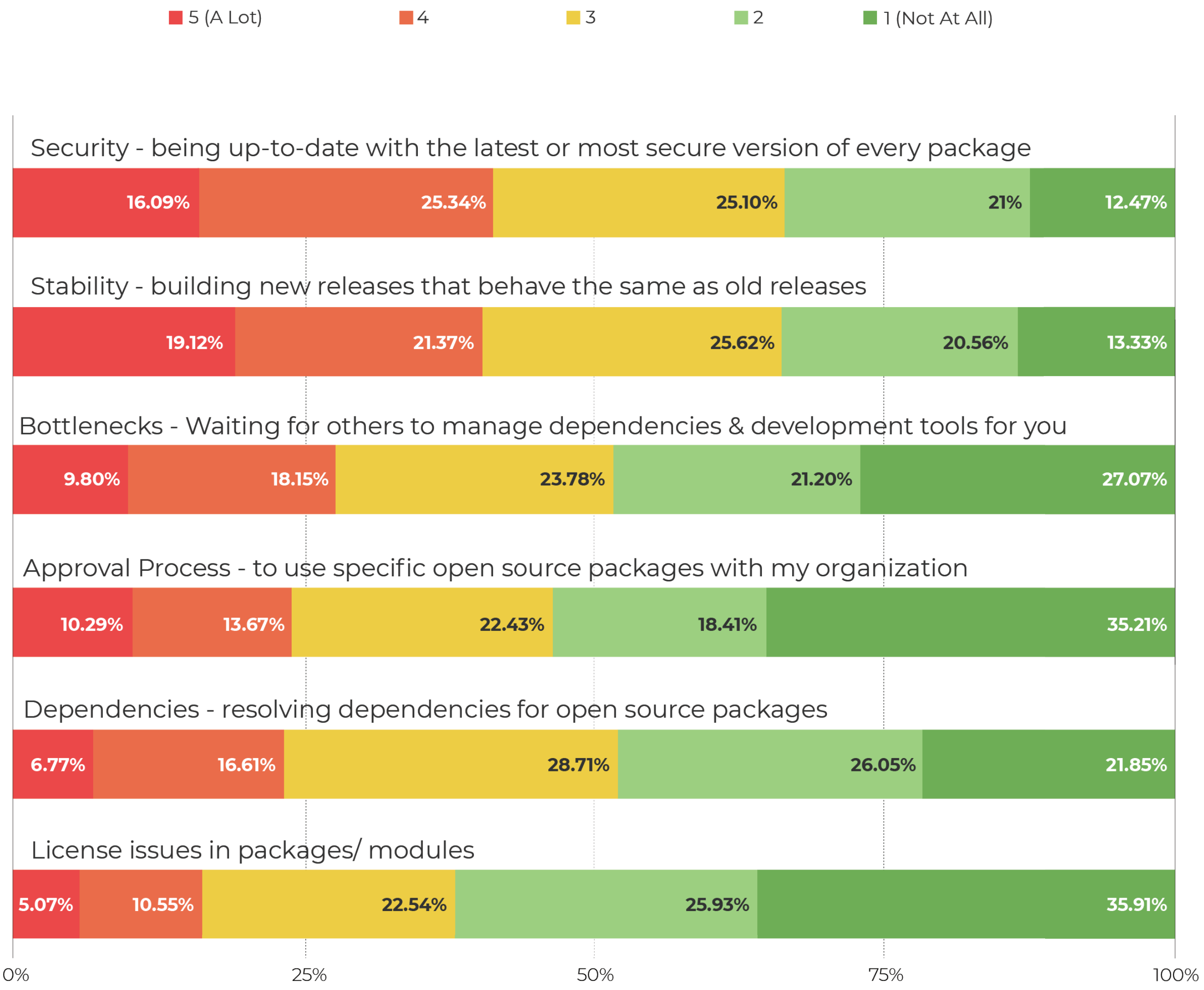
Challenge Ranking



Respondents were asked to rank the difficulty of a number of challenges from very difficult (5) to not difficult (1). This is similar to the question asked in ActiveState's 2018 Developer Survey. Additional challenges were queried to better distill the challenges faced by developers with open source runtimes. In 2018 "Adding or incorporating a language into an organization" was rated the most difficult challenge, by a significant margin. 56% of all respondents rated this as difficult to very difficult. This year 52.14% of respondents rated it as difficult or very difficult, an approximate 7% decrease. Comparatively, 16.1% of respondents rated reproducible builds as very difficult or difficult, which was similar to 2018's results, when 18% of respondents considered reproducible builds to be very difficult or difficult.

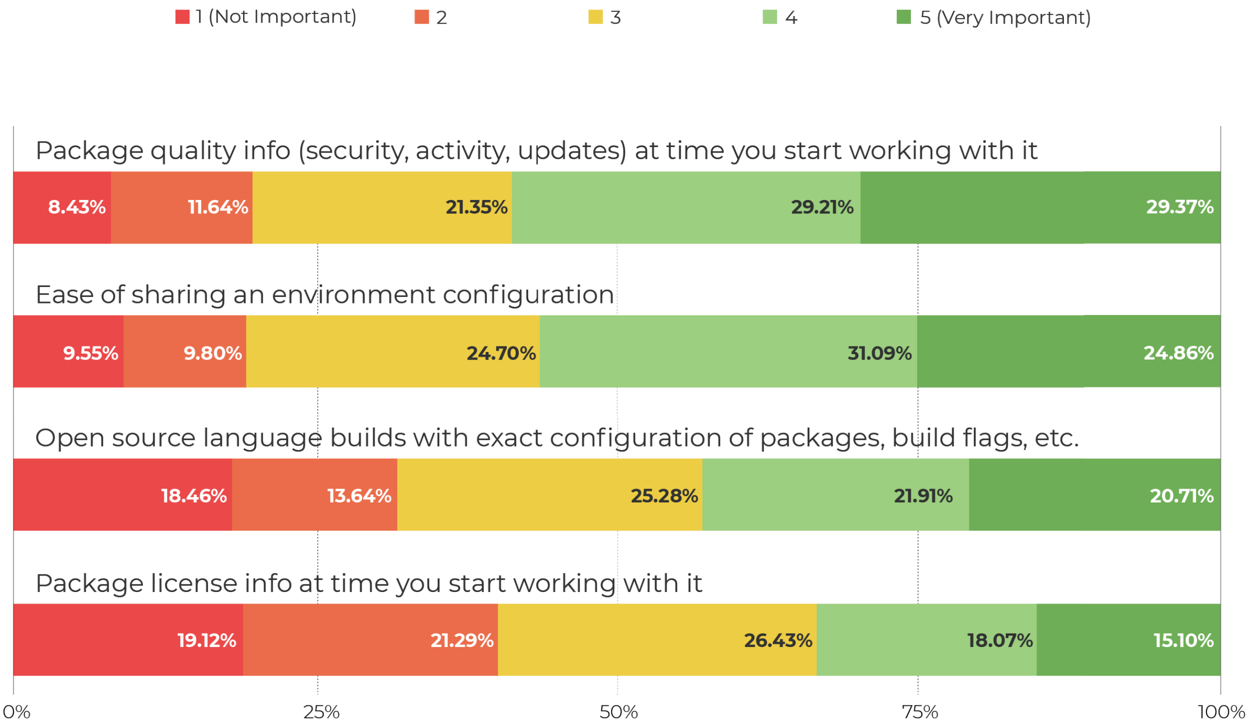
After adopting a language, security challenges was considered to be the next most challenging task with 28.03% of respondents considering it very difficult or difficult.

Build Pipeline



We also asked respondents to rank, on a scale from 5 (a lot) to 1 (not at all), whether aspects like security, stability, and bottlenecks caused problems, issues or concerns for them. Security and stability at 41.43% and 40.49% respectively, caused some or a lot of problems for respondents. Although these two items also caused the most pain points and challenges for respondents in 2018, the issues were more widespread. Last year, around 20% more respondents, 50% and 47% experienced some or a lot of problems with security and stability respectively. 2019 results for finding packages or modules for an open source language distribution had only 21.83% of respondents experiencing some or a lot of problems. This is almost identical to the 2018 results of 22%. However, i t's interesting to note that bottlenecks (like waiting for others to manage dependencies & development tools, and the approval process to use specific open source packages) ranked third and fourth, with 27.95% and 23.95% respondents categorizing it as causing some or a lot of problems.

Importance Ranking

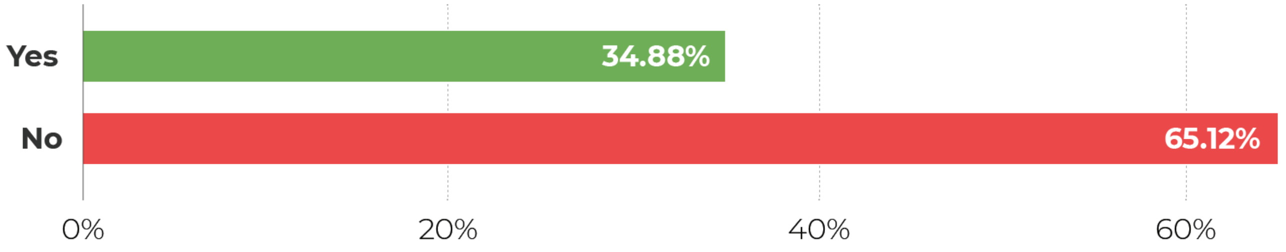


Respondents were asked on a scale of 1 (not important) to 5 (very important) to rank the importance of package quality, ease of sharing environment configuration, open source language builds with specific configuration and package license information.

Results for information about package quality and ease of sharing an environment configuration ranked close in importance by respondents, with 58.58% and 55.95% respectively considering it to be important or very important. This is slightly less important than last year's results which came in at 61% and 60% respectively.

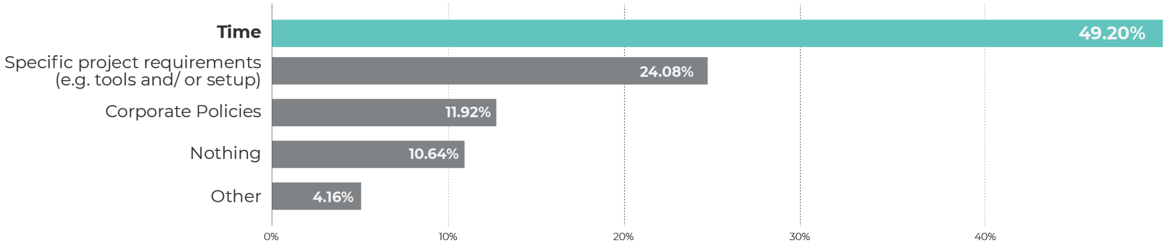
Open Source

Open Source Contributions



This year we asked respondents if they contributed to or maintained open source projects. Just over a third of respondents contributed to open source projects and nearly two-thirds did not.

Influence for Open Source Contributions



As a follow-on question, this year we also asked what would influence respondents to contribute more to open source projects. Overwhelmingly, almost half of respondents, 49.2%, selected "Time". Interestingly, almost one quarter of respondents, 24.08%, stated specific project requirements in the form of tools or setup, would positively influence their contributions to open source projects.

Conclusion

We hope the results of this survey will continue to help us understand the runtime pains faced by developers, and that these insights will enable contributions to open source communities to address key pain points developers experience with open source runtimes.

ActiveState has been working in open source for over 20 years. We've been building languages like Python, Perl, Ruby, Go and Tcl. Last year we announced our SaaS Platform to automatically build, certify and resolve open source languages...continuously and rapidly.

Interested in learning more? Want to check out what we're working on with Open Source Language Automation?

Create a free account or schedule a personalized **demo here**

ActiveState®

website: www.activestate.com | Toll-free in NA: 1.866.631.4581 | email: solutions@activestate.com

ABOUT ACTIVESTATE

ActiveState helps enterprises scale securely with open source languages and gives developers the kinds of tools they love to use. More than two million developers and 97% of Fortune 1000 companies use ActiveState open source language builds including CA, Cisco, Pepsi, Lockheed Martin and NASA. To learn more, visit activestate.com