The 2017 Top Programming Languages

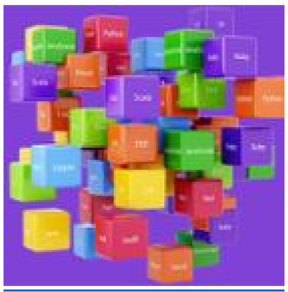
Python jumps to No. 1, and **Swift** enters the Top Ten

By **STEPHEN CASS**   Posted 18 Jul 2017 | 19:00 GMT

Advertisement

Editor's Pick

Top Programming Languages 2017: Focus on Jobs

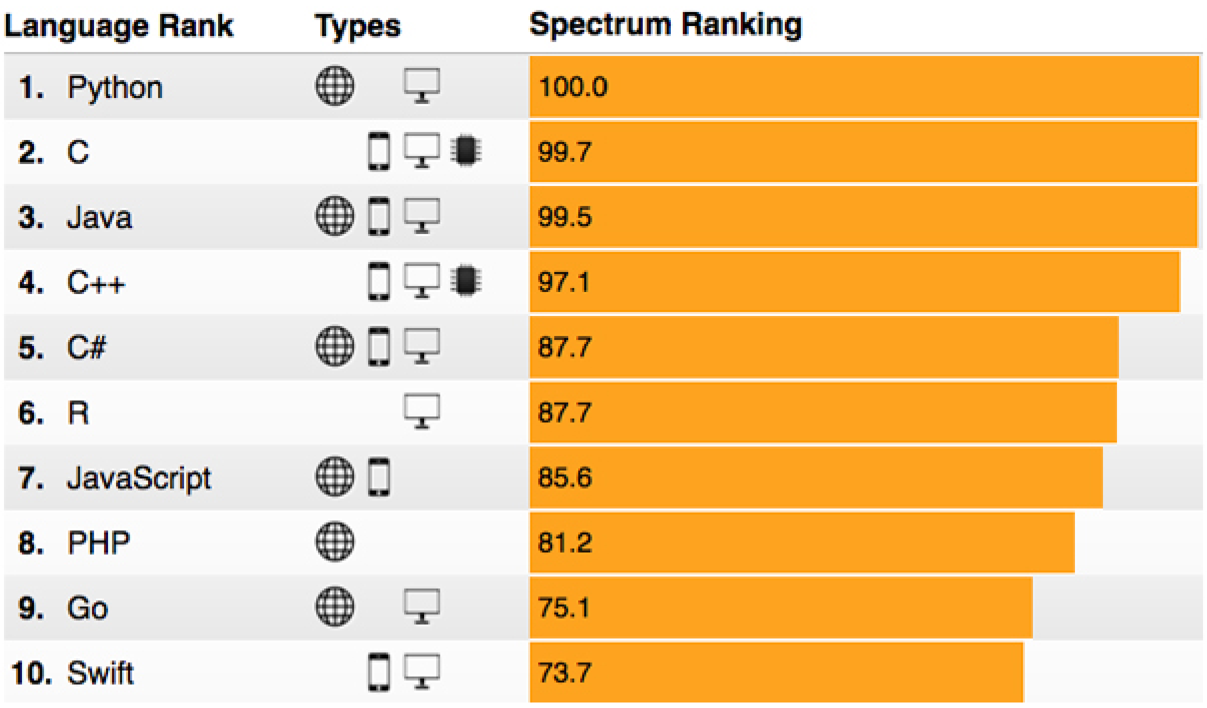


Interactive: The Top Programming Languages 2017

It’s summertime here at *IEEE Spectrum*, and that means it’s time for our fourth interactive ranking of the top programming languages. As with all attempts to rank the usage of different languages, we have to rely on various proxies for popularity. In our case, this means having data journalist Nick Diakopoulos mine and combine 12 metrics from 10 carefully chosen online sources to rank 48 languages. But where we really differ from other rankings is that our interactive allows you choose how those metrics are weighted when they are combined, letting you personalize the rankings to your needs.

We have a few preset weightings—a default setting that’s designed with the typical *Spectrum* reader in mind, as well as settings that emphasize emerging languages, what employers are looking for, and what’s hot in open source. You can also filter out industry sectors that don’t interest you or create a completely customized ranking and make a comparison with a previous year.

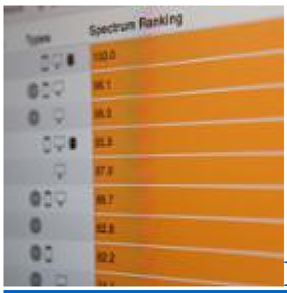
So what are the Top Ten Languages for the typical *Spectrum* reader?



Language Rank	Types	Spectrum Ranking
1. Python	🌐 🖥	100.0
2. C	📱 🖥 ▪	99.7
3. Java	🌐 📱 🖥	99.5
4. C++	📱 🖥 ▪	97.1
5. C#	🌐 📱 🖥	87.7
6. R	🖥	87.7
7. JavaScript	🌐 📱	85.6
8. PHP	🌐	81.2
9. Go	🌐 🖥	75.1
10. Swift	📱 🖥	73.7

Python has continued its upward trajectory from last year and jumped two places to the No. 1 slot, though the top four—Python, C, Java, and C++—all remain very close in popularity. Indeed, in Diakopoulos's analysis of what the underlying metrics have to say about the languages currently in demand by recruiting companies, C comes out ahead of Python by a good margin.

C# has reentered the top five, taking back the place it lost to R last year. Ruby has fallen all the way down to 12th position, but in doing so it has given Apple's Swift the chance to join Google's Go in the Top Ten. This is impressive, as Swift debuted on the rankings just two years ago. (Outside the Top Ten, Apple's Objective-C mirrors the ascent of Swift, dropping down to 26th place.)


Explore the Interactive Rankings

However, for the second year in a row, no new languages have entered the rankings. We seem to have entered a period of consolidation in coding as programmers digest the tools created to cater to the explosion of cloud, mobile, and big data applications.

Speaking of stabilized programming tools and languages, it's worth noting Fortran's continued presence right in the middle of the rankings (sitting still in 28th place), along with Lisp in 35th place and Cobol hanging in at 40th: Clearly even languages that are decades old can still have sustained levels of interest. (And although it just barely clears the threshold for inclusion in our rankings, I'm pleased to see that my personal favorite veteran language—Forth—is still there in 47th place).

Looking at the preset weighting option for open source projects, where we might expect a bias toward newer projects versus decades-old legacy systems, we see that HTML has entered the Top Ten there, rising from 11th place to 8th. (This is a great moment for us to reiterate our response to the complaint of some in years past of "HTML isn't a *programming* language, it's just markup." At *Spectrum*, we have a very pragmatic view about what is, and isn't, a recognizable programming language. HTML is used by coders to instruct computers to do things, so we include it. We don't insist on, for example, Turing completeness as a threshold for inclusion—and to get really nitpicky, as user Jonny Lin pointed out last year, HTML has grown so complex that when combined with CSS, it is now Turing complete, albeit with a little prodding and requiring an appreciation of cellular automata.)

Finally, one last technical detail: We've made some tweaks under the hood to improve the robustness of the results, especially for less popular languages where the signals in the metrics are weaker and so more prone to statistical noise. So that users who look at historical data can make consistent comparisons, we've recalculated the previous year's rankings with the new system. This could lead to some discrepancies between a language's ranking in a given year as currently shown, versus the ranking that was shown in the original year of publication, but such differences should be relatively small and not affect the more popular languages in any case.