Lizenz-Quiz

Um die Kenntnisse der Umfrageteilnehmer zum Thema Softwarelizenzen nicht allein ihrer Selbsteinschätzung zu überlassen, haben die Forscher mit den Probanden ein Multiple-Choice-Quiz veranstaltet. Hier nun die fünf Fragen, die Prozentzahlen spiegeln die Antworten der Befragten wider. Per Kreuzchen im Kästchen kann jeder Linux-Magazin-Leser selbst sein Wissen testen.

1. Welche Open-Source-Lizenz verlangt, dass so regulierter Code auf private oder akademische Software-Projekten beschränkt bleibt?
   - ☐ GPL (5 %)
   - ☐ BSD License (10 %)
   - ☐ Keine der genannten (68 %)
   - ☐ Weiß nicht (17 %)
2. Angenommen jemand schreibt ein proprietäres Programm und bettet darin Open-Source-Code ein. Welche Lizenz erfordert es unter Umständen, dass der Entwickler das ganze Programm unter die betreffende Open-Source-Lizenz stellen muss?
   - ☐ GPL (56 %)
   - ☐ BSD License (1 %)
   - ☐ Mozilla Public License (MPL) (1 %)
   - ☐ Sowohl GPL als auch MPL (27 %)
   - ☐ Keine der genannten (1 %)
   - ☐ Weiß nicht (14 %)
3. Falls Open-Source-Code ein Patent verletzt, gegen wen kann der Patenteigentümer Ansprüche geltend machen?
   - ☐ Nur den gegen den Entwickler des Codes (6 %)
   - ☐ Gegen den Entwickler und Dritte, die den Code integrieren (52 %)
   - ☐ Niemanden, weil Open-Source-Lizenzen Patente ausschließen (3 %)
   - ☐ Weiß nicht (39 %)
4. Welche der Open-Source-Lizenzen verlangt, dass solchen Code enthaltenden Produkte den Lizenztext mit ausliefern?
   - ☐ GPL (16 %)
   - ☐ BSD License (3 %)
   - ☐ MPL(1 %)
   - ☐ GPL, BSD und MPL (60 %)
   - ☐ Keine der genannten (1 %)
   - ☐ Weiß nicht (19 %)
5. Jemand veröffentlich ein Code-Snippet in einer Newsgroup oder auf einer Tutorial-Webseite. Unter welchen Umständen kann ein Dritter das Snippet rechtlich gefahrlos integrieren?
   - ☐ Wenn der Veröffentlicher keine Verpflichtungen nennt, die mit der Verwendung des Snippets einhergehen (10 %)
   - ☐ Wenn der Veröffentlicher explizit darauf hinweist, dass er keine Verpflichtungen an die Verwendung des Snippets knüpft (39 %)
   - ☐ Wenn das Snippet nicht Teil eines kompletten Programms ist (1 %)
   - ☐ Wenn alle drei genannten Bedingungen erfüllt sind (16 %)
   - ☐ Keine der genannten Bedingungen reicht aus (19 %)
   - ☐ Weiß nicht (15 %)

Wer die Auflösung des Rätsels wissen will, knickt diese Seite unten rechts diagonal nach innen und kriegt einen Kasten mit den richtigen Antworten zu Gesicht.

ihre Arbeit ein (siehe **Abbildung 1**), nur 12 Prozent bezeichneten es als „überhaupt nicht wichtig" während für 19 Prozent Reusing sehr „sehr wichtig" war.

Fügt man der Betrachtung durch Berücksichtigen der ehemaligen Entwickler eine zeitliche Dimension hinzu, ist erkennbar, dass das Thema über die Jahre bedeutender wurde (siehe **Abbildung 2**). Umfrageteilnehmer, die bereits vor 2004 aufgehört hatten Software zu entwickeln, bewerteten Open Source meist zwischen „überhaupt nicht wichtig" und „kaum wichtig". Erst nach 2003 begann die Wichtigkeit, bis sie schließlich 2008 und 2009 eine durchschnittliche Relevanz von „Ein wenig wichtig" erreichte. Dies fällt zeitlich mit der Verbreitung von Open-Source-Software zusammen.

Neben dem Zeitverlauf scheinen Erfahrungen und berufliche Stellung die Einstellung zum Thema zu beeinflussen. So lässt sich feststellen, dass professionelle Softwareentwickler, die über eigene Erfahrungen in Open-Source-Projekten verfügen, solchen Code auch professionell bedeutsamer einschätzen. Möglichweise tun sich solche Entwickler leichter damit, passenden Code für ihre Projekte zu finden und zu evaluieren.

Interessanterweise beeinflusst auch die eigene Tätigkeit im Unternehmen die Haltung zu Open Source. Softwarearchitekten und Projektmanager zeigen sich gegenüber der Wiederverwendung von Code aus dem Internet aufgeschlossener als Programmierer. Deren Einschätzung wiederum liegt höher als die von Testern, Datenbankentwicklern und Systemanalysten. Dass sich die drei zuletzt genannten Rollen wenig mit dem konkreten Implementieren beschäftigen, macht die geringere Bedeutung plausibel.

Der Unterschied zwischen Softwarearchitekten und Projektmanagern auf der einen Seite und Programmierern auf der anderen mag daran liegen, dass die erste Gruppe über die Beeinflussung der Architektur der zu entwickelnden Software über höhere Freiheitsgrade beim Einbinden von existierendem Code verfügt als Programmierer, für die die grundlegende Architektur der Software, an der sie arbeiten, bereits von außen vorgegeben ist.

Als Drittes zeitigt die primär verwendete Programmiersprache einen Einfluss. Befragte, die hauptsächlich mit Ruby oder Python arbeiten, erachten wiederverwendeten Code als besonders bedeutsam. Dies mag daran liegen, dass es beide Sprachen besonders einfach machen, fremden Code einzubinden, sogar solchen aus anderen Sprachen. Perl, Javascript, Java und PHP bilden eine zweite Gruppe, erst dann folgen mit C und C++ die traditionellen Sprachen. Entwickler, die mit Fortran, Visual Basic, C# oder Pascal arbeiten, legen den wenigsten Wert auf Code aus dem Internet. Kulturelle Unterschiede scheint es hingegen nicht zu geben, die Entwickler aus Europa, Nord- und Südamerika oder Asien messen quelloffenem Code etwa die gleiche Bedeutung zu.

Hier knicken!

Kaum offiziell unterstützt

Angesichts der insgesamt hohen Bedeutung der Wiederverwendung von Open-Source-Code in kommerziellen Entwicklungsprojekten ist es erstaunlich, dass der Großteil der befragten 2009 aktiven Softwareentwickler zu diesem Thema nie offiziell geschult worden ist. Auf die Frage, aus welchen Quellen sie sich zu