State of The CF Union 2017

Survey Analysis

TeraTech

Hello and welcome.

In this comprehensive 2017 State of the CF Union survey report, we have detailed breakdowns of tools and techniques used by the top CFML developers around the world. This year we had 472 responses to the survey, which is slightly lower than the last year.

Before we begin with the results, we'd like to thank everyone who participated and responded to this year's survey. And a special thanks to Brad Wood who helped me edit the survey questions and reviewed the results with me.

If by any chance you are new to ColdFusion, it is a development platform for creating modern web applications on the JVM. The CFML language has tags that resemble HTML syntax for templating HTML and script that resembles JavaScript syntax for writing business logic. It is designed to be powerful, expressive and easy to get started coding in. Many features are built into ColdFusion that require add ons for other languages.

Brad pointed out that it's not just CFML developers and enthusiasts that read through all of the survey results, CIOs and other managers read it too. And that even Adobe pays attention to the survey as well. So it's good to know that the com-

Michael Smith

CEO **TeraTech**

Brad Wood

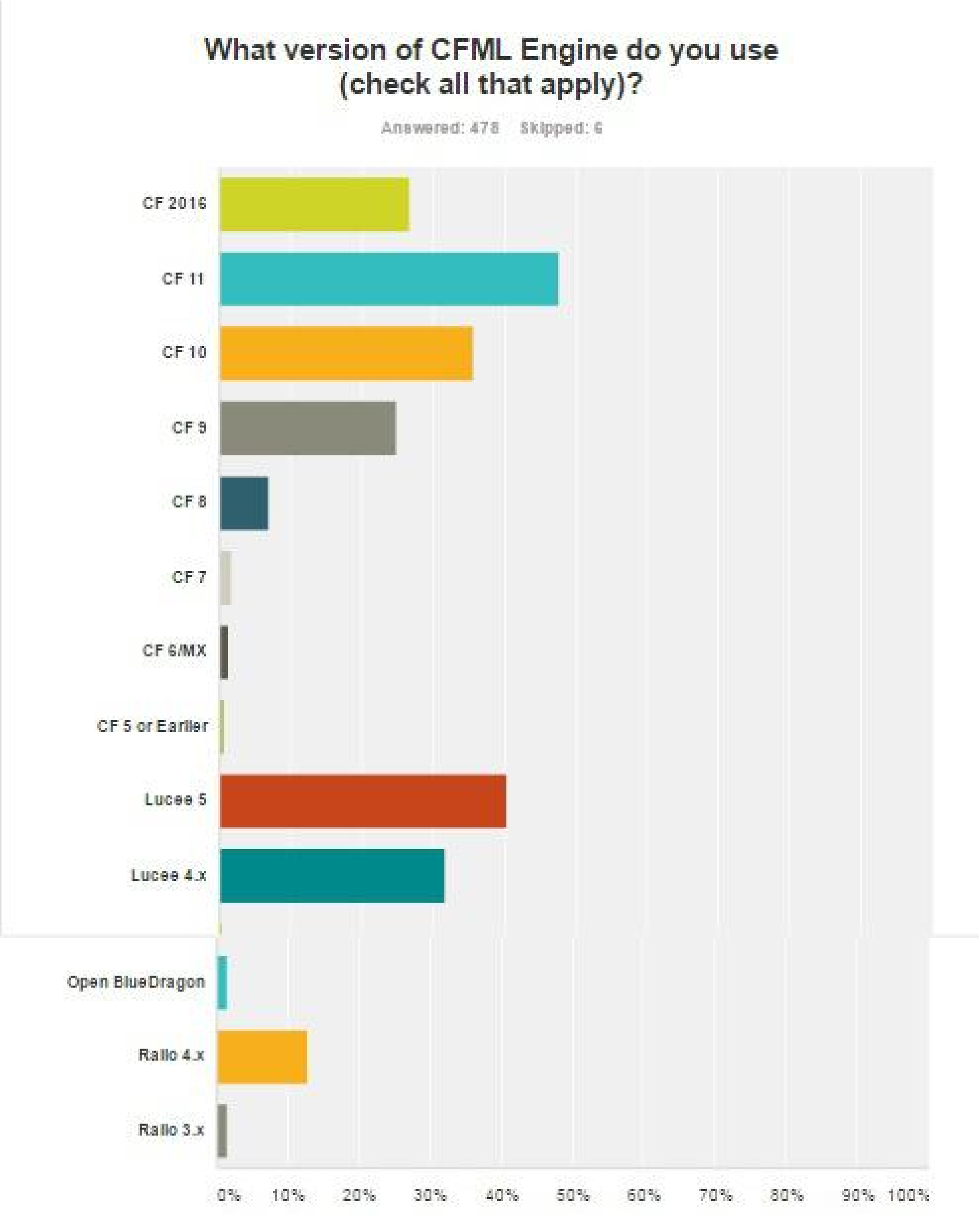
ColdBox/CommandBox Developer Advocate. Lucee Member. System Architect. CFML dev. DBA. Dad. Handyman. Ready for a revolution.

You can reach him on Twitter

Table of contents:

01. What version of CFML Engine do you use?

The most popular engine is still CF 11, which was true last year.



It's nice to see the ColdFusion 9 members finally shrinking. It was a version that a lot of people got stuck at for many years, so we feel better about dropping support for that in the ColdBox framework, ColdBox 5, for instance, coming up. CF 11 is still very much on top. 2016 is growing, but it's even behind CF 10 at this point, which is very interesting to see.

Also, there's some interesting numbers here too- Lucee 5 is actually more popular among the survey takers than CF 10 or CF 2016.

It's important to note that this is only out of the people that take the survey, so I do think the people more active in the community that are likely to see the survey are more likely to be using Lucee, but it is very heartening to see a very solid uptake, especially even just in Lucee 5. Of course, now that it's been a year or so you can see the Railo shrinking into oblivion since it's essentially unsupported at this point.

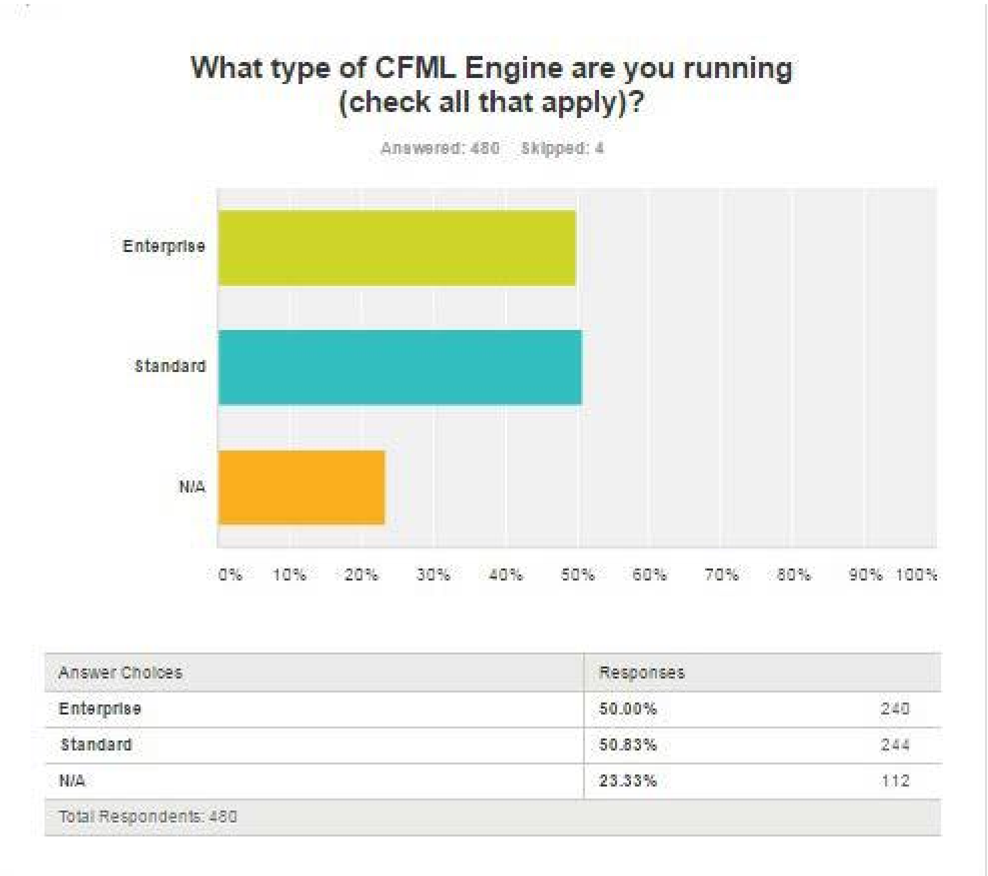
It's interesting to note as well, thinking about these graphs, which versions are still supported.

- Railo at this point has no support.
- Lucee 4 has support only for security fixes, but it's not seeing enhancements, that's all going to Lucee 5.
- Adobe stack, CF 9 was still a decent chunk of users, that's already outside of the support window, and
- CF 10 is coming right behind.

And CF 9 went end of life on 12/31/14, over a year ago now. That means no more security fixes.

CF 10 is just a very short period of time away (5/16/17), so very soon there's going to be quite a chunk of Adobe CF users that are outside of the support window, which is a little scary to think about. That means not just the support, but also no more security hot fixes, is like driving a car with no brakes and praying that nothing comes on the road that you need to stop for. (see Adobe Lifecycle table for all versions)

02. What type of CFML Engine are you running?

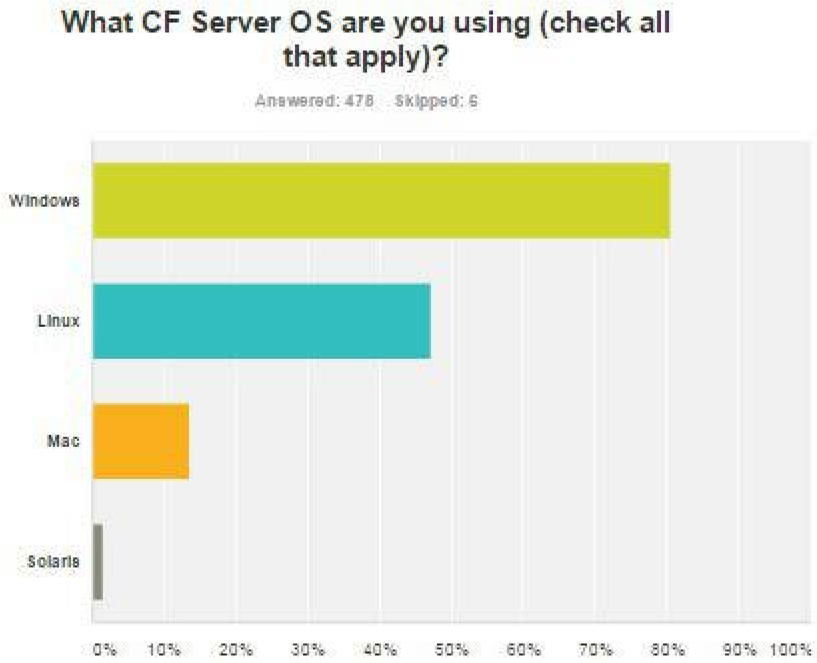


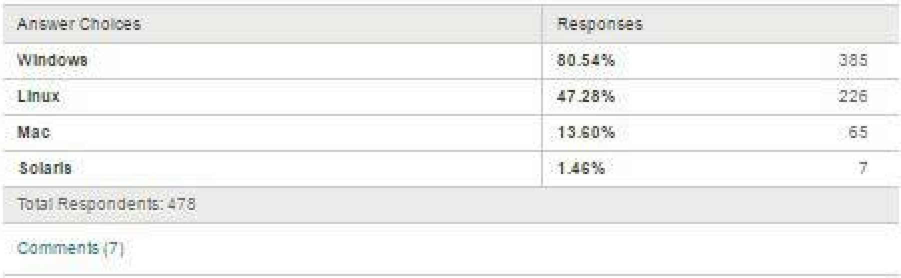
Answer Choices	Responses	
Enterprise	50.00%	240
Standard	50.83%	244
N/A	23.33%	112
Total Respondents: 480		

There's not much to say about ColdFusion Enterprise versus ColdFusion Standard, quite a lot of people using Enterprise. They are pretty *neck and neck*. The assumption is that the non-applicables are probably mostly the Lucee CFML user base, since there's no differentiation there. The reason is that you get all the features in the one version. It's either free in one version or free in the other versions. They probably decided to just go with one.

Interesting note is that back in the early Railo days, there actually used to be a premium version of Railo that had additional features, but that's no longer the case with Lucee.

03. CF Server OS





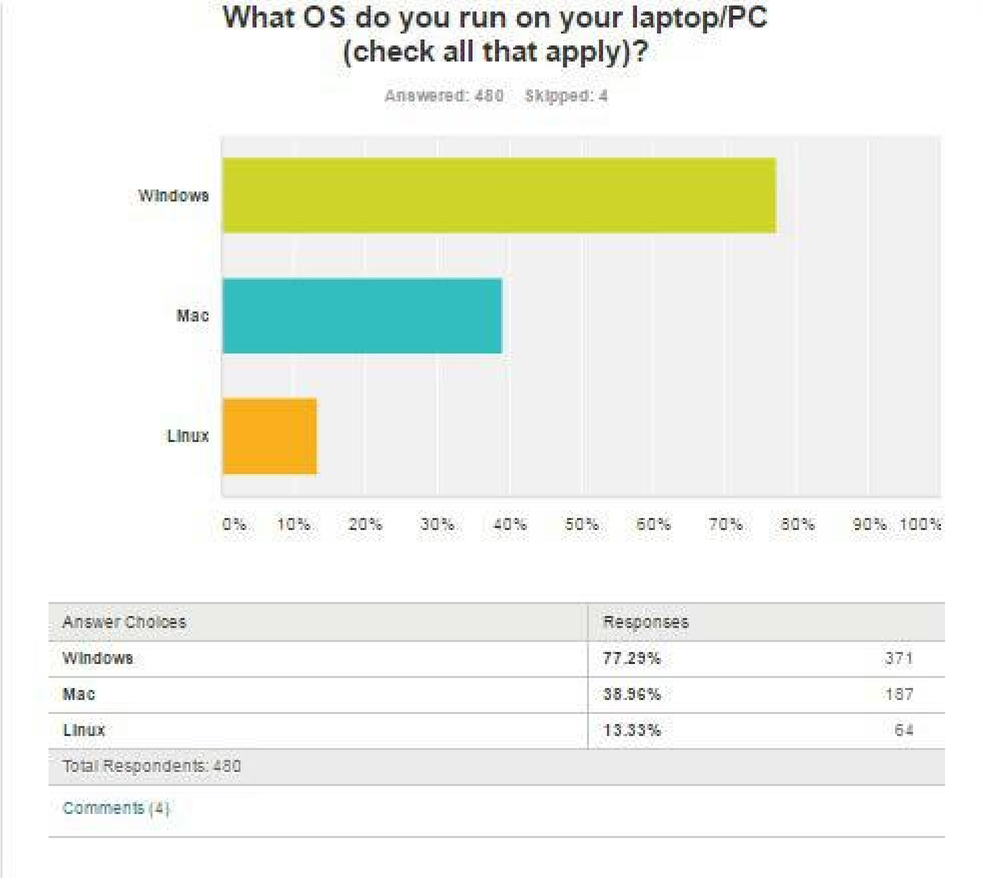
Answer Choices	Responses	
Windows	80.54%	385
Linux	47.28%	226
Mac	13.60%	65
Solaris	1.46%	7
Total Respondents: 478		
Comments (7)		

We can see that most people are using Windows but there's a big chunk of people using Linux or something similar. There's still some Unix flavors out there.

The large Windows use is something that's sort of unique to the ColdFusion and .NET environments probably. If you were to do a similar poll looking at PHP or some of the other popular languages, you'd see an inverse of a lot more Linux users. What's interesting though is there is a solid chunk of Mac people, and this is for the server OS. This means people are hosting their servers on Macs, and that's actually a decent chunk.

Sometimes ColdFusion doesn't work exactly the same on Mac, but maybe they're running a staging server or development server, and they develop on Mac and they're hosting it on a Mac.

04. What OS do you run on your laptop/PC?



What OS do you run on your laptop/PC (check all that apply)?

Answered: 480 Skipped: 4

Answer Choices	Responses	
Windows	77.29%	371
Mac	38.96%	187
Linux	13.33%	64
Total Respondents: 480		

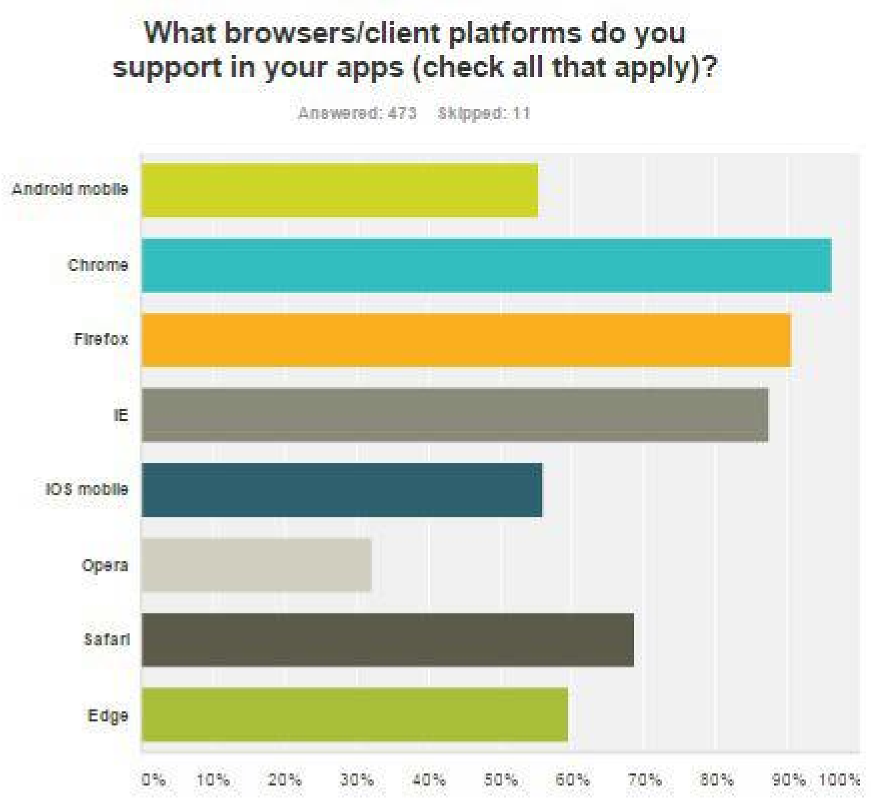
Comments (4)

Mac and Linux is obviously going to flip-flop here. This question is particularly interesting for Brad as a lead developer of CommandBox. It's a native CLI tool that runs on the operating system, so this helps to know what to target as far as the operating systems people are going to be running those tools on. This also plays in as well with the operating systems that Adobe supports for things like CF Builder.

There's a lot of reasons to have your laptop on Linux. It's more secure and you've got more options and things, and you can still emulate Windows in it if you need to. Lot of folks go that way.

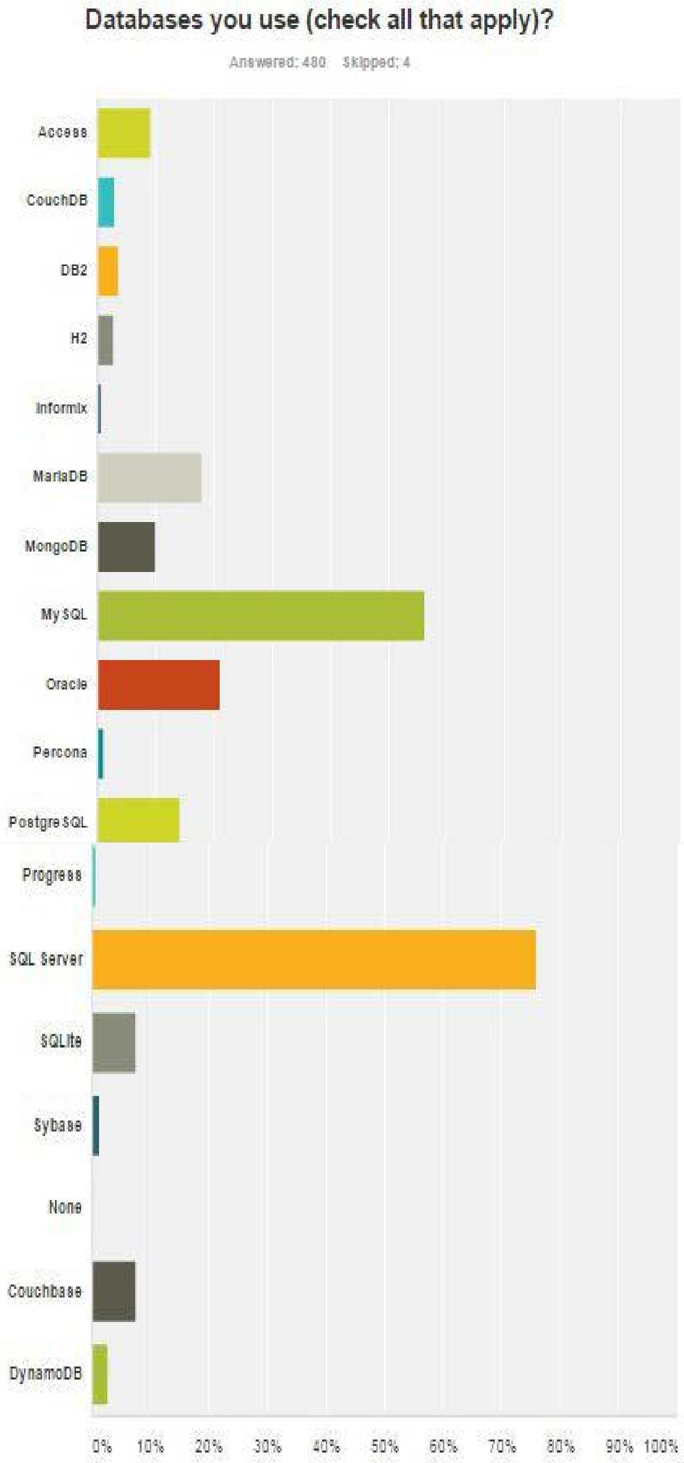
05. What browsers/client platforms do you support in your apps?

There aren’t many places that explicitly don't support a browser. The only except typically being a government type of intranet site in which they'll say, "We only support IE, screw everybody else." Generally speaking, if you have users with money in their hands saying, "Shut up and take my money," then whatever browser they're using, your company's in favor of supporting. It's just harder to be a place that won't support a particular browser.



Microsoft got a little fat and lazy for a while. Maybe they stepped up a bit in the last few years, but there seemed to be a while where they really weren't updating IE as much as Google updated Chrome.

06. Databases



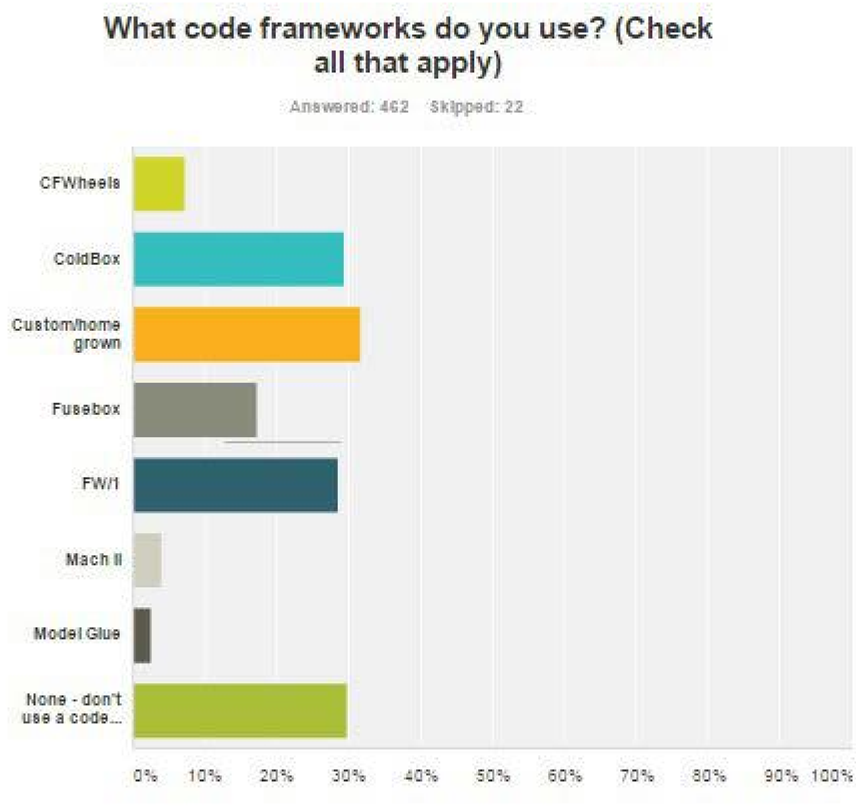
Everyone uses a database, right? Some of us use Access, apparently.

Probably everybody who does ColdFusion since the late '90s used Access at one point in time. It's fine for developing, as long as you get it into MySQL or SQL Server or something else later. Because Access can not deal with higher loads of a production site.

There's nothing incredibly surprising here. CF and SQL Server is a common pairing, and a lot of the government uses a ColdFusion, especially with Windows-based hosting. MySQL is probably the most common source. However, it is little surprising that PostgreSQL isn't a little higher.

Oracle's pretty big too, and it is nice to see some other options like Couchbase things with some decent uptake in there. Couchbase has a distributed architecture. It started out as a marriage of Memcached and CouchDB, but it's grown quite a bit since then.

07. Code frameworks



What is the most popular framework? There's a lot of people still to fall on the custom homegrown framework or they don't use a framework at all.

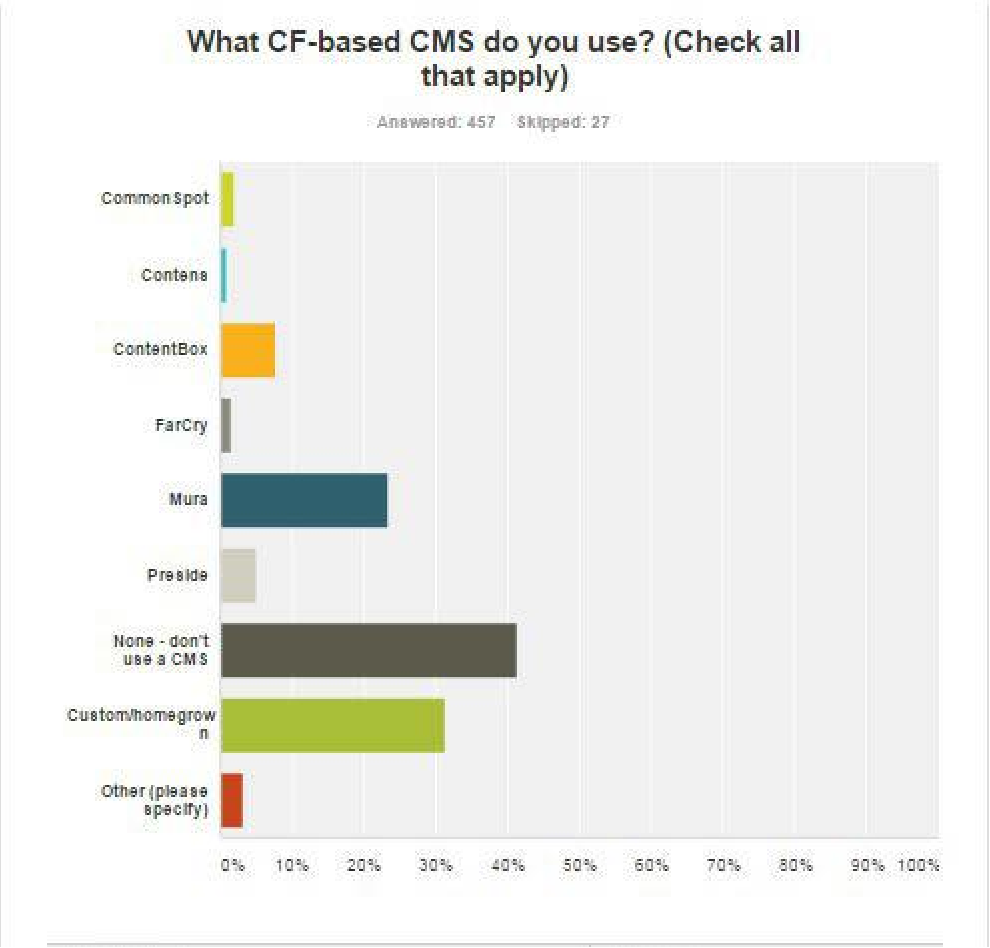
A very large percentage of the ColdFusion code out there falls in the legacy bucket, and even if those people would use a framework today, they're unlikely to rewrite their from scratch to use on now. Those two bars haven’t fallen much, to be honest. The ColdBox and the Framework One bars increase every year a little bit, but it seems like the other ones don't manage to decrease too much.

Even Fusebox has a pretty solid usage of 18%, even though not too many people are building new things, but it's just a lot of existing FB legacy sites.

We'd better declare a winner here, which is ColdBox by a hair's breadth there. Looks like there's a few more ColdBoxes than FW1s, but they're both very popular. And then Fusebox respectable, and then Mach-II, Model-Glue, and CFWheels is at least the third framework in that list that is actively developed on, whereas Mach-II and Model-Glue have essentially hit the end of their road.

08. CF-based CMS

CMS - the content management systems.



We broke this out into a separate question this year. It used to be mixed in with the framework question, and it makes more sense to look at CMSs (Content Management System) separately.

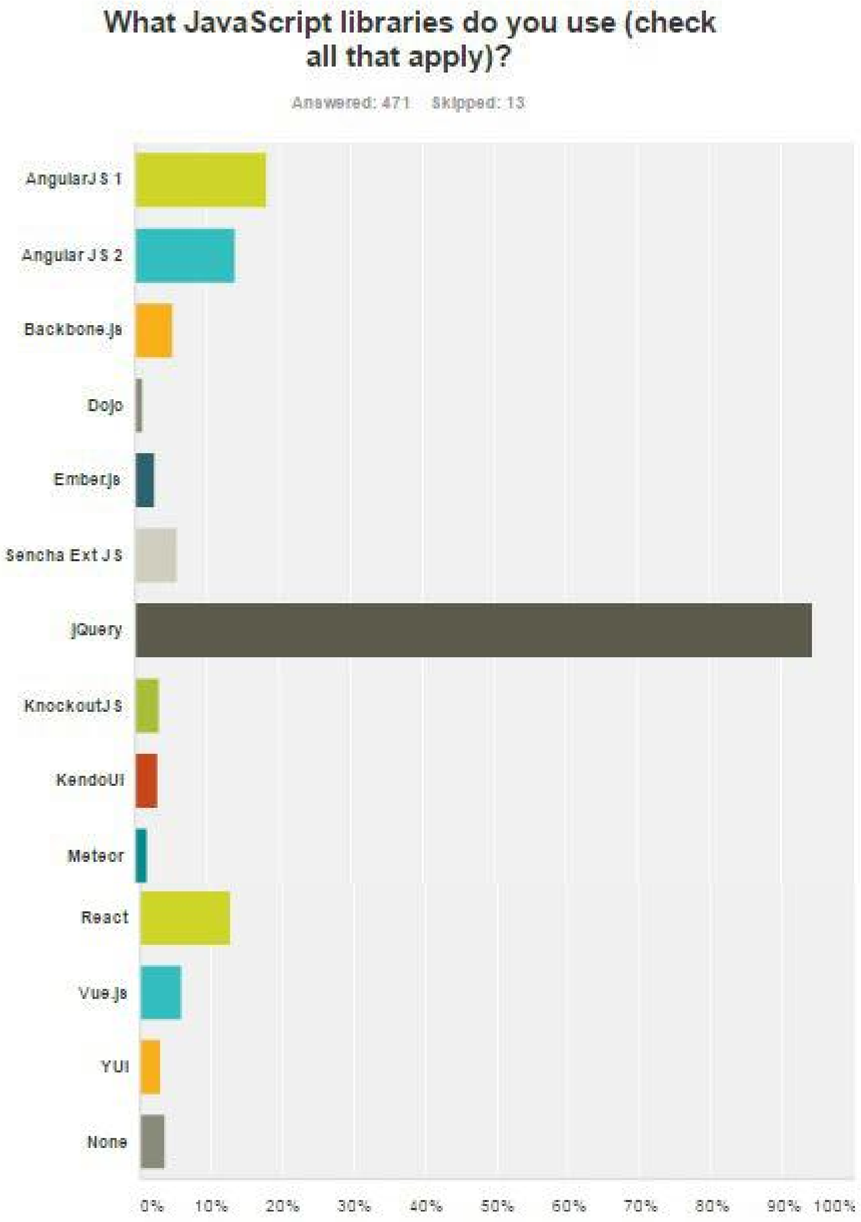
The custom homegrown, there's a lot of homegrown CMSs in the ColdFusion space, which shouldn't be that surprising, but it's interesting.

Mura is the clear winner. They're probably the oldest ColdFusion CMS.

FarCry is very old too, but there isn't anyone outside of Daemon using it. Geoff Bowers helps run that.

And, Mura of is open-source, so you can tweak with it however you feel. Interestingly, the winner is custom homegrown.

09. JavaScript libraries



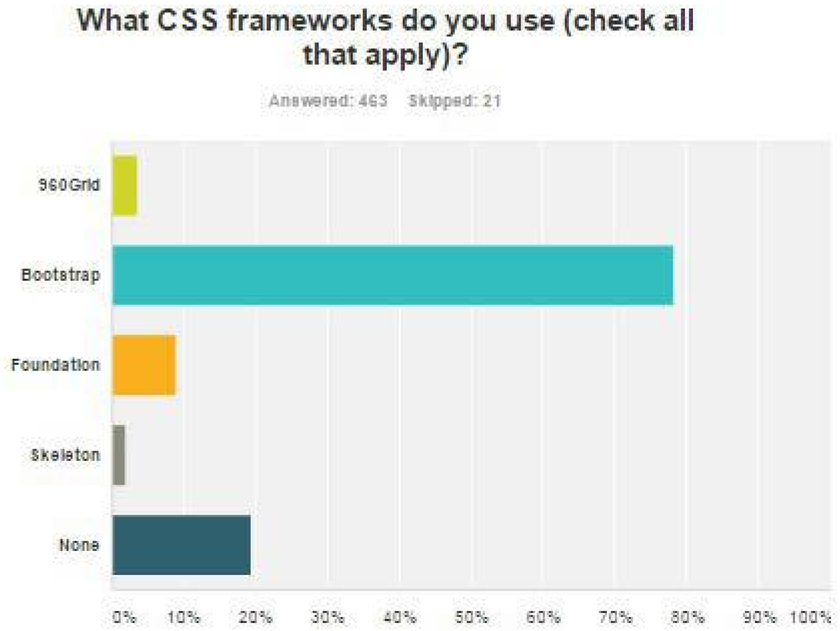
jQuery is used by nearly everyone.

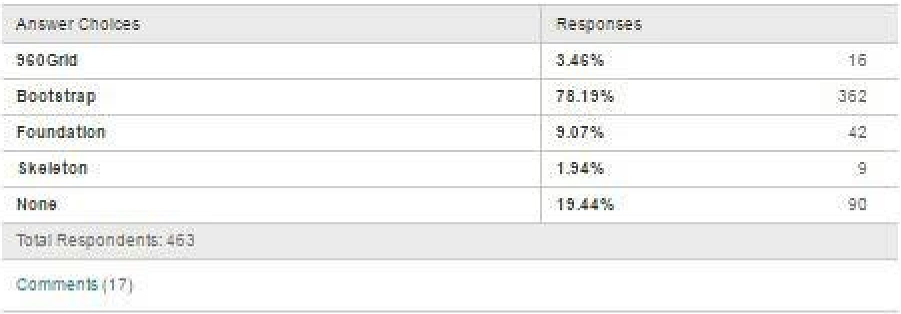
Brad considers jQuery more of a building block than a comprehensive library like Angular and such. It's no surprise to him that everybody uses jQuery.

You can see that Angular 1 is still ahead, but there's a lot of uptake in Angular 2. React (at the bottom of the list) is the next closest thing that we have there. All the others are down in the 5% kind of zone.

There's a lot of choice out there. In the survey we only covered 15 of about 15 billion JavaScript frameworks that exist right now.

10. What CSS frameworks do you use?

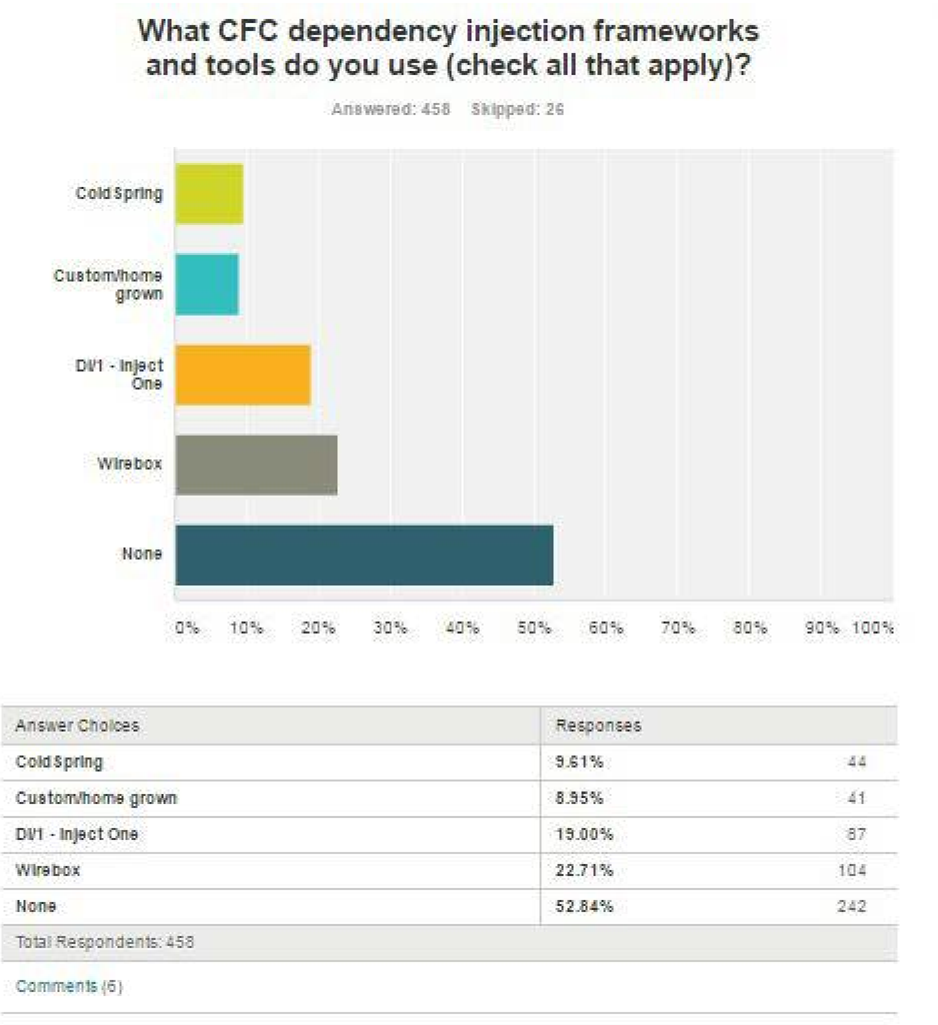




Answer Choices	Responses	
960Grid	3.46%	16
Bootstrap	78.19%	362
Foundation	9.07%	42
Skeleton	1.94%	9
None	19.44%	90
Total Respondents: 463		
Comments (17)		

Bootstrap's the winner there, but it's not like everyone uses Bootstrap. There's definitely use of some other frameworks there, and using no framework is also popular. Total response is 459, so most people did respond to that question.

11. What CFC dependency injection frameworks and tools do you use?



What CFC dependency injection frameworks and tools do you use (check all that apply)?

Answered: 458 Skipped: 26

Answer Choices	Responses	
Cold Spring	9.61%	44
Custom/home grown	8.95%	41
DI/1 - Inject One	19.00%	87
Wirebox	22.71%	104
None	52.84%	242
Total Respondents: 458		

Comments (6)

CFCs dependency injection. Just in case some people aren't clear why they'd even want to use it, maybe we'd better explain what that actually is. Dependency Injection (DI) frameworks help you create your CFC instances and they help inject references to other CFCs to help manage the relationships between them.

There's a much larger segment of the ColdFusion community that says they're using CFCs, but that's not really reflected here in the who's using dependency injection. It could be that a lot of people using CFCs more in a very simplistic level. It's just manually creating their own CFCs and not getting too fancy.

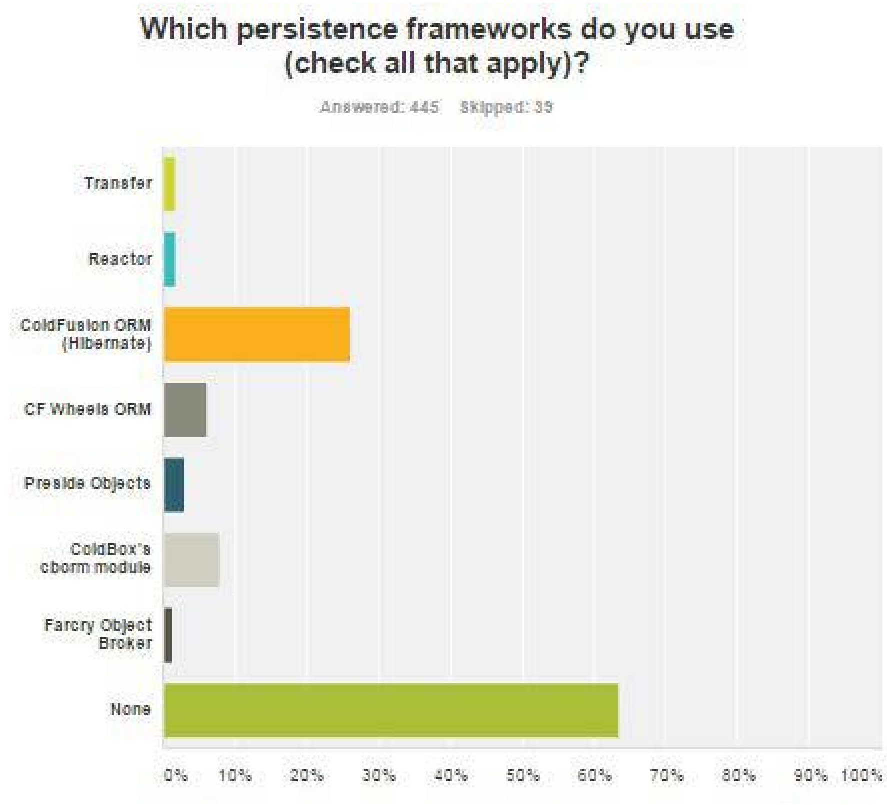
If you only have two or three CFCs and a few functions in each, that probably doesn't matter, but once you start getting hundreds of the things, it really can get screwed up if you don't do dependency injection.

If you're using CFC seriously, you really want to be using one of these dependency injection frameworks.

WireBox and DI/1 are actively maintained.

- WireBox obviously part of the ColdBox ecosystem
- DI/1 part of the Framework One ecosystem
- Cold Spring has been a project that hasn't seen development in many years, but there still seems to be a decent amount of use out there

12. Persistence frameworks



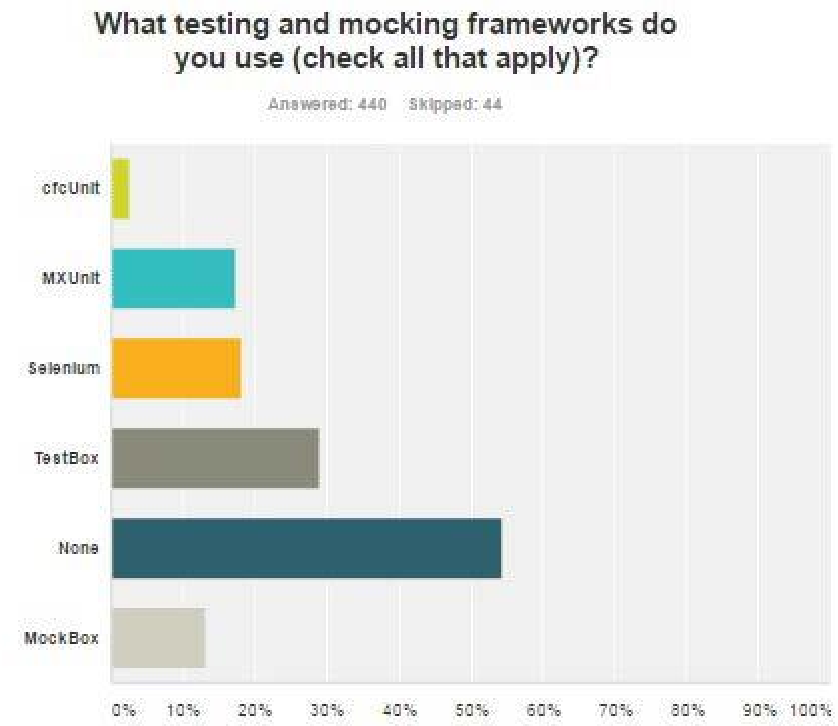
It looks like most people don't use ORM, but if you are using objects in your code and a system framework to instantiate them, the one built into ColdFusion is what most people are using. And these aren't all mutually exclusive. For instance, the ColdBox cborm module is actually a library you use along with ColdFusion ORM, not as opposed to, so that's interesting. You can see a lot of the major frameworks, the CMSs, have their own versions, CFWheels.

A lot of those were written in last year in the same questions, so we added them as possible responses for this year's survey. Transfer is probably the closest that the ColdFusion community ever saw to the full ColdFusion ORM.

ORM isn't an incredibly popular option in the ColdFusion community. A lot of places like Ruby and Groovy, Grails, they really push ORM as just the default way to built things. It's never quite caught on as much in the CFML space.

In Michael's opinion, people like their cfquery tags and don't want to let someone else's code manipulate their data, but can speed up development if you do the ORM approach.

13. What testing and mocking frameworks do you use?



What exactly is mocking? Let's take Brad's explanation.

''When you start writing unit tests, you want to test a single function in isolation, only the code in that function. If that function calls another function, you want to mock and have a fake function call, but it can call for you. Mocking is just a library that creates fake CFCs, fake functions, fake queries to fake a database call, basically used for creating mock data and mock interfaces so you can eliminate everything else but the code you're running the test on.''

And then presumably automate the testing, so once you've made a change, the tests will get re-run automatically.

MockBox is a part of TestBox. You don't have to use MockBox, but it comes a part of TestBox. TestBox is the only maintained testing framework at this point. MXUnit was once very popular, but the creators no longer maintain it.

Selenium is a browser-based testing, like a web driver, though there is a CF Selenium project.

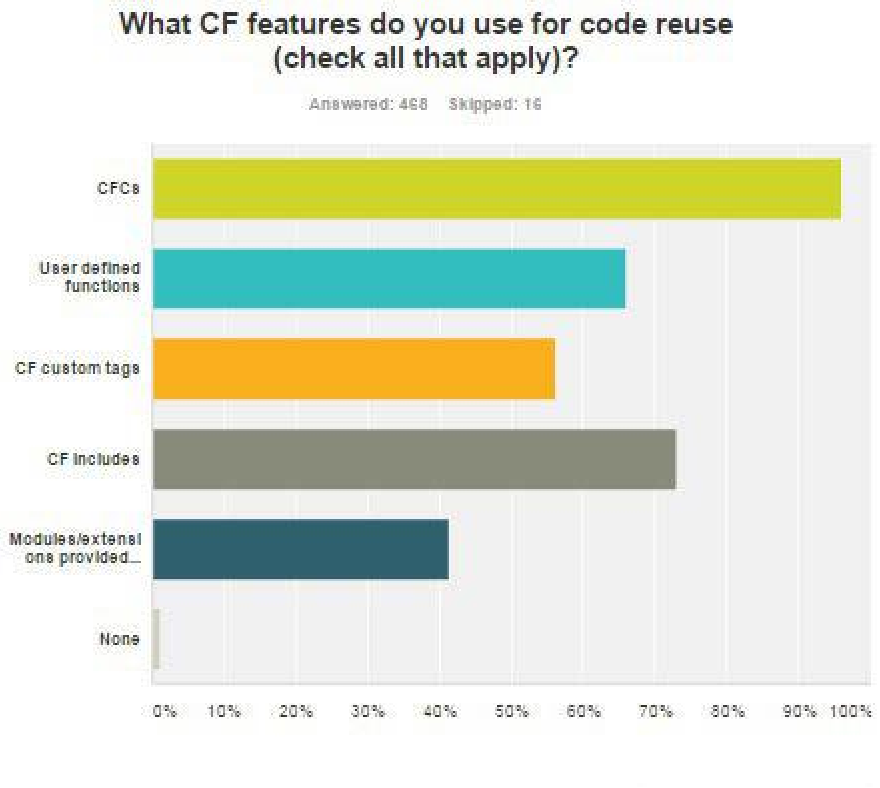
And, MXUnit was based on a Java unit testing library.

If you've got some tests written in MXUnit, you can just pull them into Test Box. TestBox actually has a compatibility feature where it will run any MXUnit test out of the box. In fact, a couple notable people, Sean Corfield and Adam Cameron, when TestBox first came out had a very large MXUnit test suite, and they just ported them right over to TestBox with very few changes. It was done on purpose to make it easy to switch. No one wants to go through and rewrite all of your tests, right?

Adobe and Lucee both do their best to be backwards compatible. Sometimes people wish they'd be a bit less backwards compatible, but at the end of the day they get money when people upgrade, and people upgrade when it's easy, right?

Who wants to go through thousands of lines of code just because you upgraded the server? What I'd like is the old code runs fine but there's some way to get a list of things that really should be changed that are deprecated.

14. What CF features do you use for code reuse?



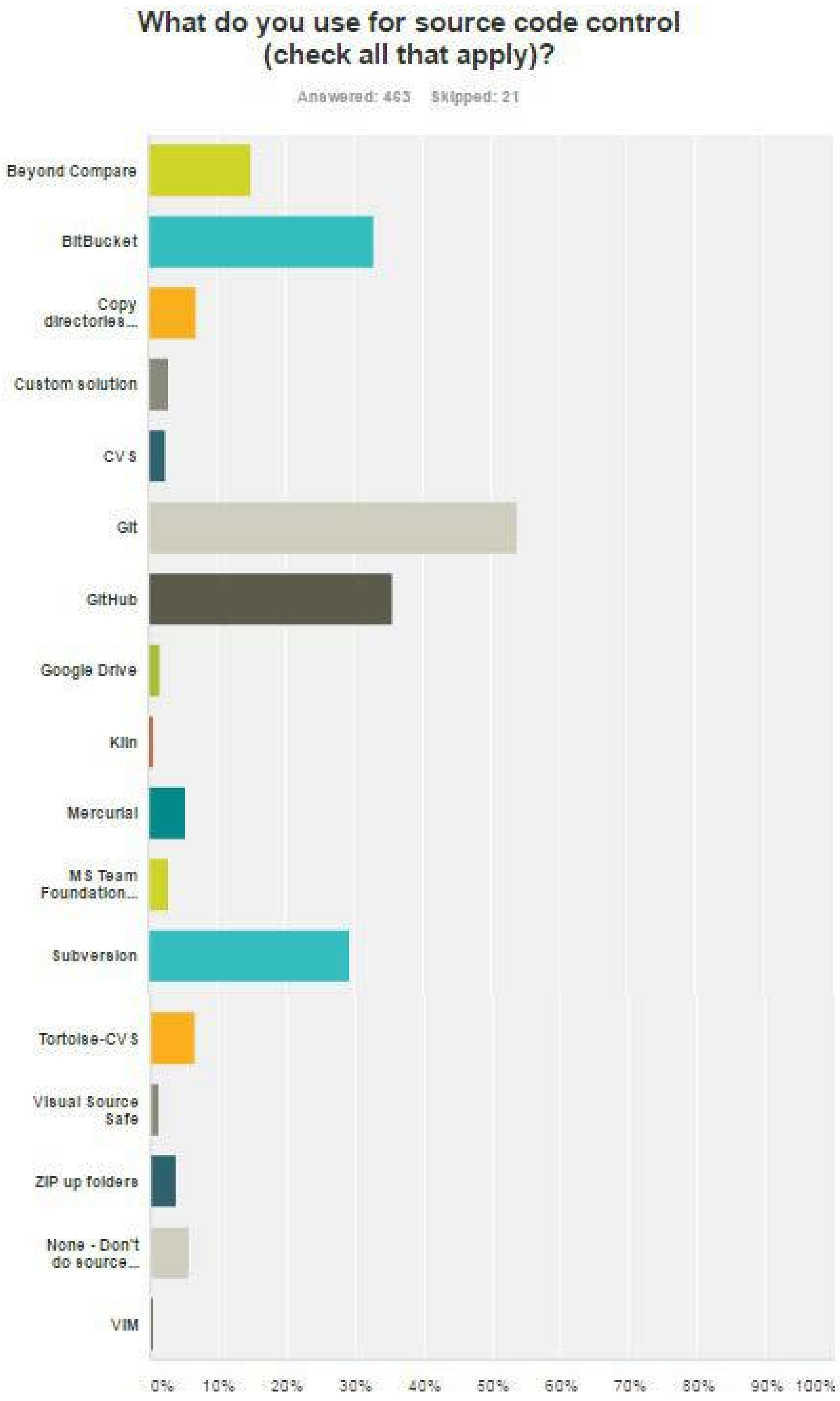
The only thing that's surprising about this survey question though is *the none*. There's actually a couple people that somehow managed to not use anything. Five people. How do you not use even cfincludes? It is surprising, unless there's five people that no longer program on ColdFusion who took the survey.

95% of people say they're using CFCs in their code, but that number is significantly lower, it is 40% that are using a dependency injection framework to manage it.

And, they're probably not really using objects. They're using them as includes in a CFC. Just a piece of code they're calling and they're not really dealing with the object stuff. Or, it could be they're doing all this and just didn't know you should be using dependency injection.

People have apps in production that don't use dependency injection and it solves business problems.

15. What do you use for source code control?



This was somewhat contentious last year. A certain Mr. Cameron wrote the blog post that ripped a few butt holes for people who didn’t do “real source control”.

One of the first times the survey was run, the "I don't use source control at all," question, which was here at the bottom, was up to like 20% maybe. It was a very large amount of people.

Still got 5% there. It's still pretty popular, but you've also got the zip-up folders option, which were done 15 years ago.

Also, If we assume that people who copy directories, zip them up, or don't do anything are all separate, that's 15% of people still not really doing source control in any serious way.

That has shrunk, which is good. But I would like to see it at 0% next year!

Git is emerging as the clear winner here.

Again, this is check all that apply question, so people who check Git could also check GitHub. Bitbucket as well. There's actually a bit more Bitbucket than expected which is pretty interesting.

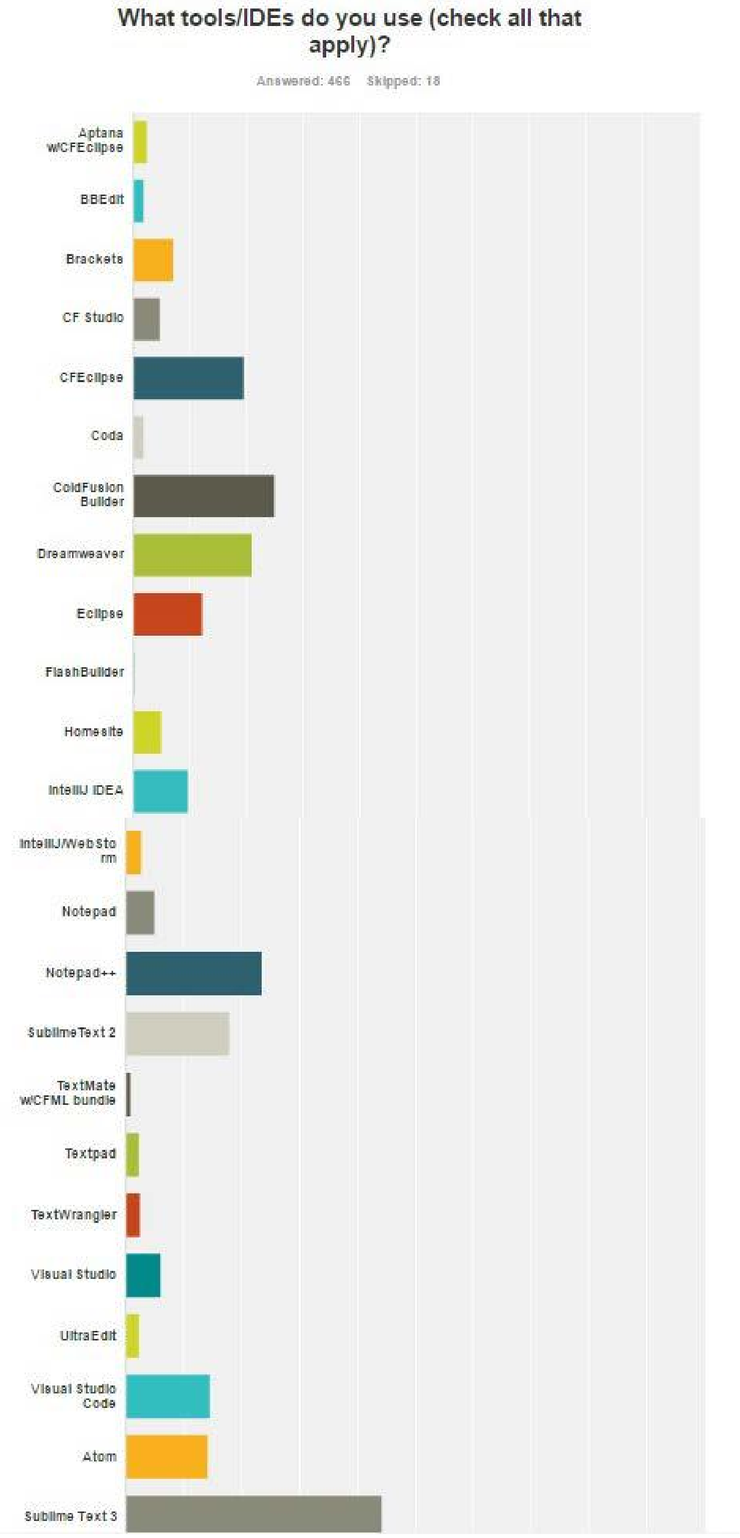
Bitbucket seems to specialize more in private repos, where GitHub seems to specialize more in public. Bitbucket of course also allows Mercurial repositories as well as opposed to. Some people using Bitbucket may be using Mercurial, whereas everybody who checked GitHub in theory checked the Git box as well.

And then if you're not into Git, there's always Subversion, which is still popular. 10 years ago I think it probably would've had the most votes in here.

But there's a lot of advantages to get if you're working with people in different locations and want to branch more easy.

The open-source community has really embraced Git in general as the source control to use, especially just with GitHub becoming the hub for open-source projects. It makes a lot of sense.

16. What tools/IDEs do you use ?



IDEs and tools, always a good place for a healthy discussion among developers. And CFers are no exception.

It was surprising to see a lot of CFEclipse users, and that's great to know that people are still getting a lot of use out of it. Denny Valiant has been doing some maintenance on that, so there's still more people than expected. Even Dreamweaver, having dropped ColdFusion support for a while and then now very recently bringing it back, there's still a lot of people using presumably old versions of Dreamweaver.

Even Notepad++ has a lot of people. That editor has an add-on that supports CFML. There's some code coloring, so for a quick and dirty editor it's not too bad.

We removed a couple text editors that had low responses last year. Like Vim used to be on there and maybe Emacs. We did that because we only had like two or three hits last year, so we just removed both.

When scrolling to the bottom that it looks like the most popular IDE is Sublime Text 3. And that's no surprise. It's by far what most everybody in the room is using. There's ColdFusion highlighting, there's ColdBox plugins, there's a CommandBox plugin, there's a lot of community input around Sublime Text 3, so it's no surprise to see it being used so heavily. It’s 44% and Adobe CF Builder at 25%.

CFEclipse, which is older, 19%. Notepad++ is beating them at 23%.

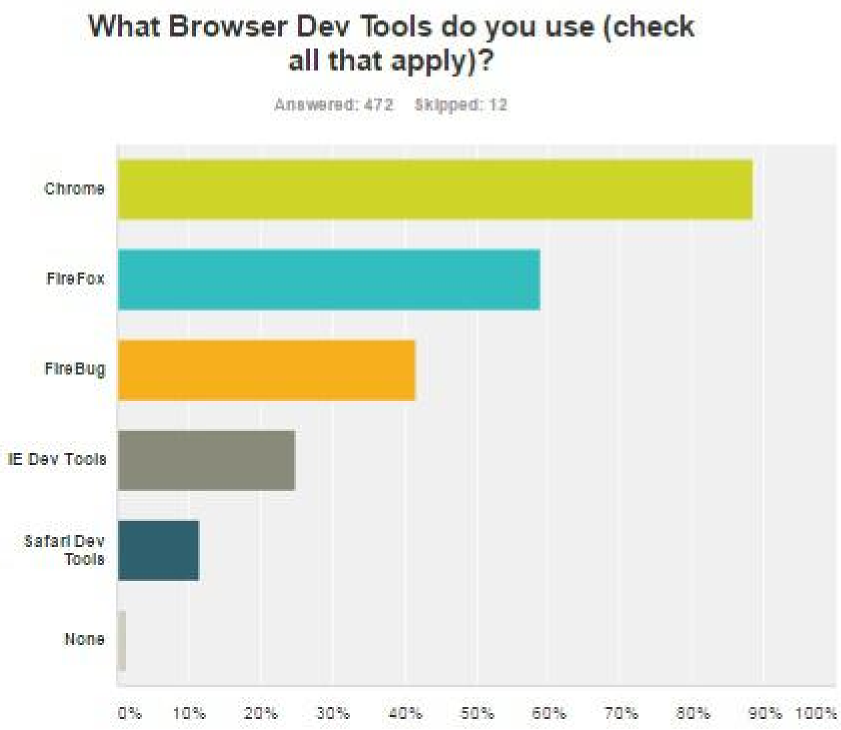
This was a check-all, so it's entirely possible people have one of these on their server, one of them on their local machine.

What happens with IDEs, is that you learn one and you get to like it and it does it good enough, and you just stay with it.

One of the biggest reasons people change is plugin communities and speed. Several of these on the list are all Eclipse-based, which is based on Java, it can be slower, can take a lot of memory. One of the big things about Sublime is it's always been very lightweight and very fast, and that's attractive to a lot of people. They don't want an IDE that's very heavy, it locks up for five minutes while it re-indexes everything or whatever it might be doing.

Some of these other ones, like Ultra Edit is really fast and TextPad and Notepad++ probably have the same benefit. They're not running on Java, they're coded in C probably. They just run a lot faster, even if they don't have all the amazing bells and whistles that you might get in CF Builder.

17. Browser Dev Tools

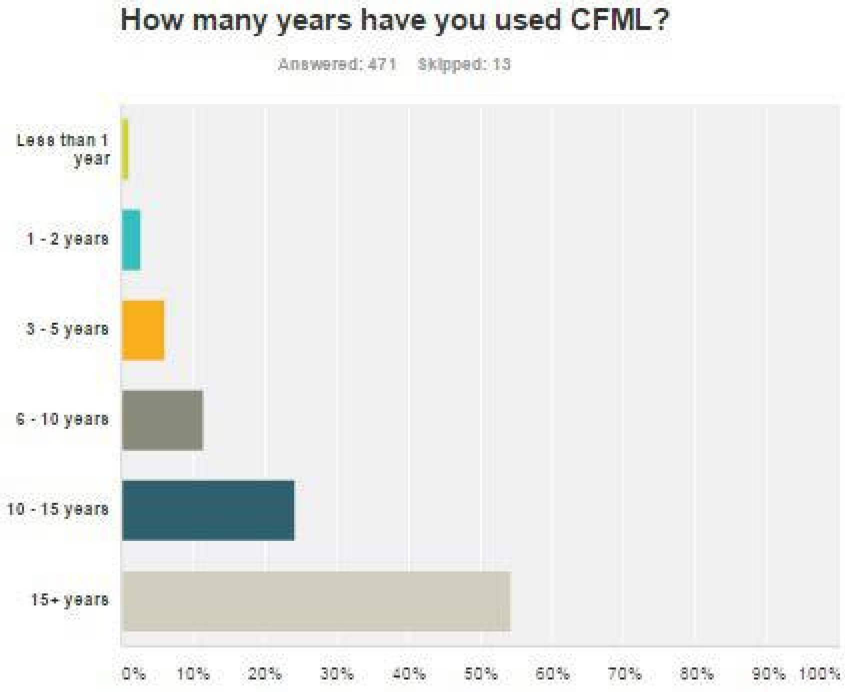


Most of the users are mostly using Chrome and only fire up Firefox when Chrome is feeling sick for the week.

Back in the day, when Firefox was busy smashing Internet Explorer, it was the golden child for a while, and then Google started leaning their weight on Chrome when it became very popular.

What are you going to do if you have an open-source project like FF and then Google decides they're going to have a competing one? You're got a big problem…

18. How many years have you used CFML?



It's very clear that a lot of people have been doing CF for a long time, basically since ColdFusion 1.5.

This is one of the more sobering questions on here, because this isn't really what you want to see in a language. If you look at Clojure, which is a language that has about as many people doing it as ColdFusion but it's only a few years old, you'll see a graph that's almost the complete inverse. First of all, it doesn't go to 15 years because it hasn't been around that long, but second of all, you'll see the majority of people using it all started in the last one or two years.

Some of the Java developers say that the Java community suffers from a lot of the same sort of stuff, lots of legacy code, lots of older people just age-wise in general as well as the amount of time they've been into it, and ColdFusion is not as new, it's not as cool, not as many people flowing in.

This is something that the wider ColdFusion community, that Lucee, that Adobe all need to work on addressing. How do we get new people in to look at the CFML language?

Where are the new people coming into the language? They're just not there, at least not for the people answering the survey.

Eventually you're going to have just a natural attrition. All the people in that 15 year group are going to retire at some point, right? They're not going to develop until they're 99 years old.

Retire or get run over by a bus, one or the other.

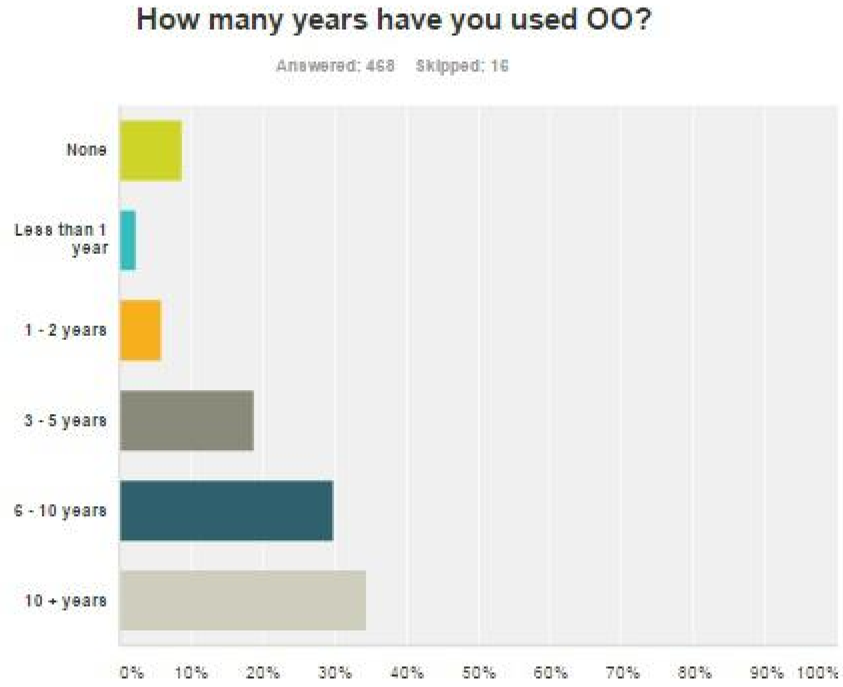
Brad said that this is an area of opportunity.

‘’I think this is best serviced by grassroots. I know a lot of people say they want to see Adobe do more, they want to see Lucee do more. They always want people to do more and that's fine, but I think this kind of growth happens at the grassroots level. I think people start using Clojure because they run into people at a local, open source user group that say, "Man, I'm using this language called Clojure. It's amazing, you should try it." I think that's the kind of stuff that grows these people looking into a language even more so than Adobe running marketing ads somewhere. I guess I say that to say the onus is on all of us as ColdFusion developers to help fix this graph.’’

Do some mentoring of some students in a college or teach a course in a college, just encourage younger people to give it a go because it's so easy to get going on it and it's really powerful, because we don't want to end up looking like the COBOL curve, which is even more extreme version of the same graph.

The average age of web-based technologies is actually around 22 to 23 years. If you look at Java, Ruby, JavaScript, HTML, CSS, SQL, which is like 45 years old, the average language being used to write web applications is actually a bit older than ColdFusion. We're below average, but that's not to detract from the fact that the language is very mature. Food for thought.

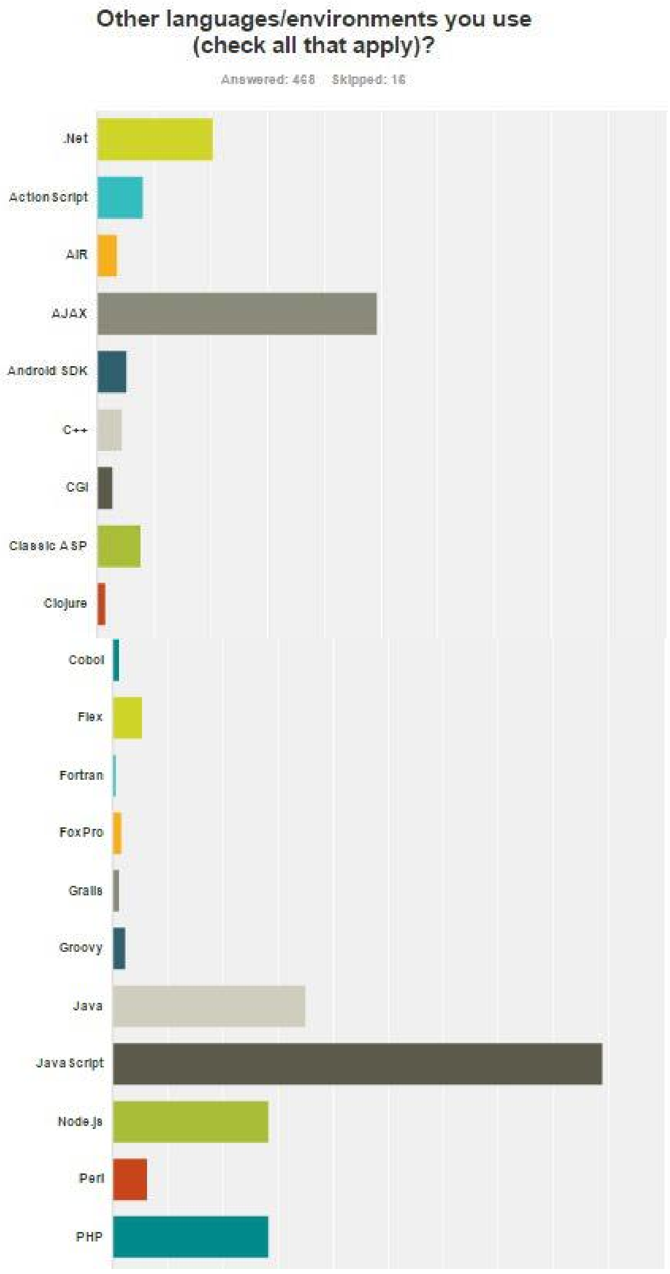
19. How many years have you used OO?

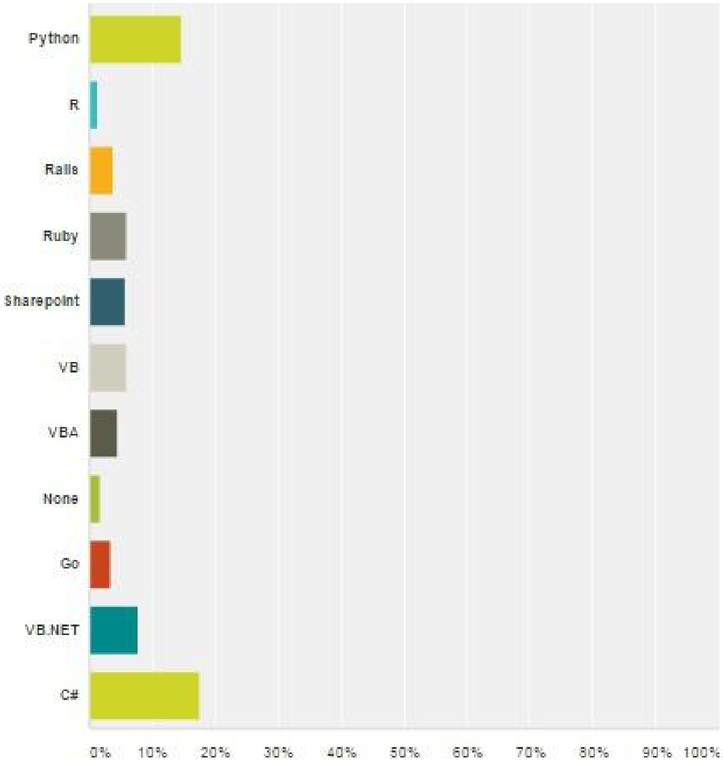


About a third of people been using it for six to 10 years, a bunch of people using it for more than 10 years.

A question here is always what are they doing with it? Are they building a modern CFML application? Or is it, "Yeah, we've got some CFCs and we put some code in it," which may solve their business problems but it may not necessarily be a modern OO application life cycle.

20. Other languages/environments you use





A lot of ColdFusion people code in other languages too.

Ajax isn't so much a language, but everybody uses Ajax, right? .NET's a big one.

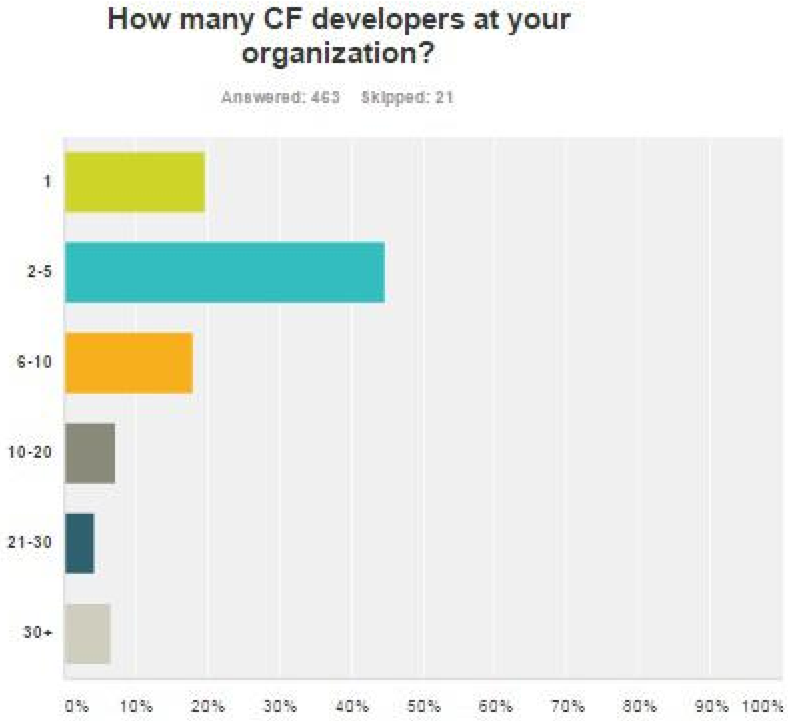
And, everybody uses JavaScript. If you're doing web development, you can't avoid JavaScript. Lot of Java, quite a bit of Node. It was expected to see a bit more Ruby.

- Rails of course is actually a framework, an MVC framework.
- Lot of Java,
- Lot of PHP
- Python
- There's a fair amount of .NET.

But really a really wide array of different languages people know. Which is something for CFers to be proud of.

Look at AIR and FLEX. There's still a decent chunk of people, and that's not too surprising. The companies that put a lot of investment into FLEX and AIR, are still using some of those things. Some people that make a lot of money updating FLEX applications.

21. How many CF developers at your organization?

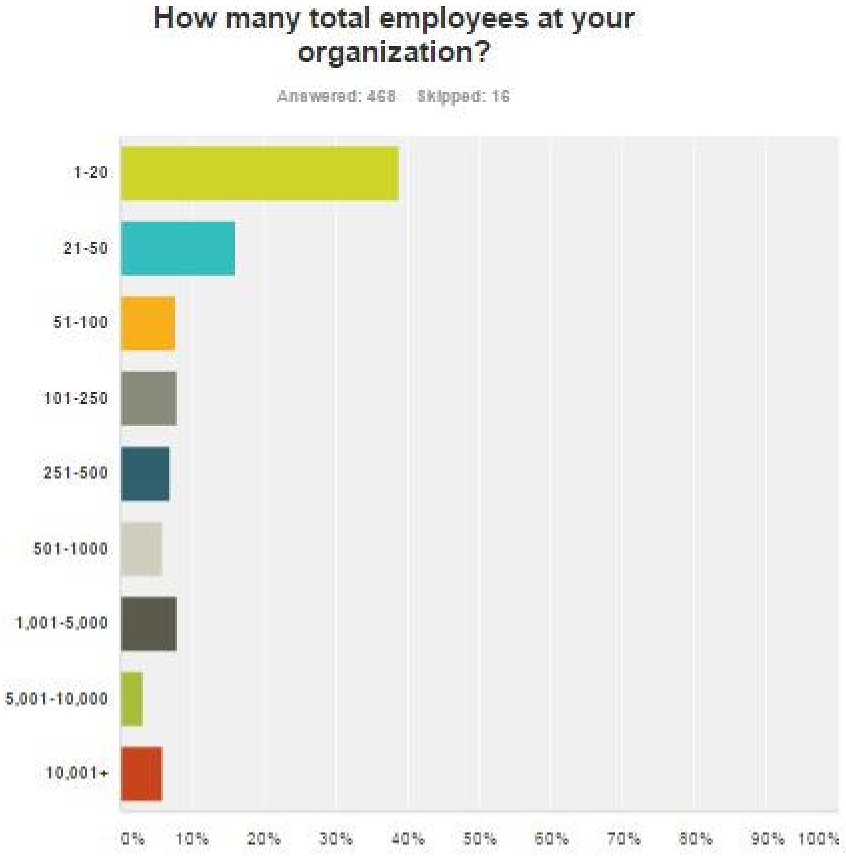


It seemed that this trended towards the corporate enterprise stuff, but this shows that there's a lot of smallish teams. The Adobe guys, Alisha and Kishore, they said they see the same thing, a lot of smaller ColdFusion teams out there. The team of one is about 20% of people ie they work by themselves.

We forgot to put zero in here for those There are some people who no longer use ColdFusion who answered the survey anyway because they were feeling generous. This means that there are some 0's also.

There's some big teams out there too. Although we've just put 30 plus here there are couple people that have more than 50 developers.

22. How many total employees at your organization?



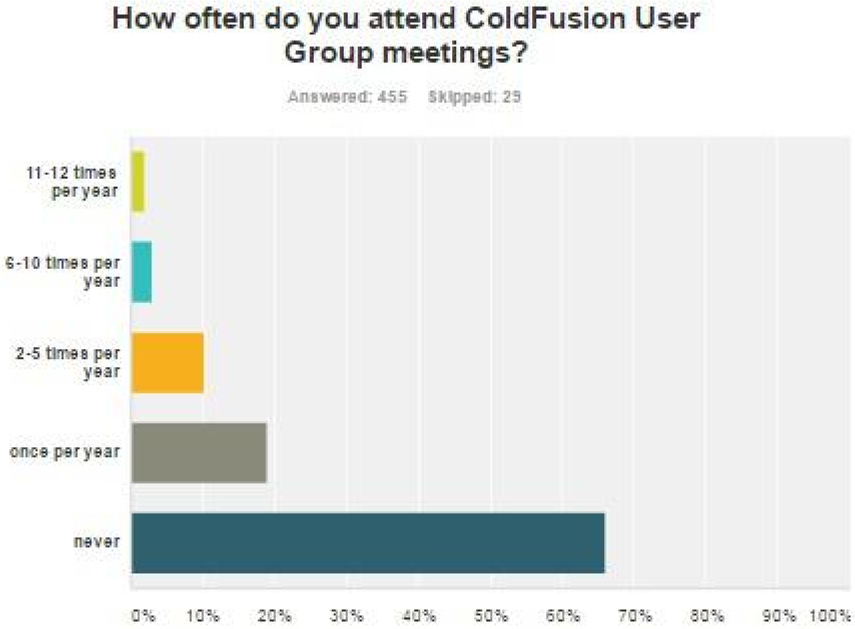
Also, iIt's a question of team size versus how many developers there. Some of the organizations, like Boeing that has a lot of ColdFusion servers and a lot of developers.

5% of people doing the survey are organizations with 10,000 or more people in them, whereas you can see the majority of people are in small organizations.

It is also refreshing to see that the majority of people taking the survey are in small business. To Brad that's kind of cool, because we talk about how ColdFusion is still only prevalent in government and the financial sector.

This is either Fortune 500 or government, this 10,000 plus. And probably most of those people who did 10,000 plus are part of some government or a small handful of Fortune 500 companies.

23. ColdFusion user group meetings



Everyone used to go to ColdFusion user group meetings ten years ago, but now, a lot of ColdFusion user groups have closed or they can't get speakers. One possible answer is that people are bored of going to them.

If you wanted to go to a user group meeting at this point, you couldn't find one if you wanted to, at least not an in-person one.

Maybe there's some pent up demand that's not being addressed.

When Michael started the MDCFUG user group, only a handful of people would turn up, but it builds over time. Even when only a handful turn up, still one can learn stuff and share info and have a good time.

It would be interesting to see how many cities are there out there, medium cities, whatever, where there's at least five or 10 people that would go to a ColdFusion user group if one of them would take the time to be able to reach the rest of them and say, "Hey, let's start up a group," because many people might think, "Well, who would come to it?" So they don't do it.

I would be interesting to know who's not going to user groups because they don't want to, like, "Yeah, I could, but I don't care," and who's thinking, "Man, I'd love to have a user group to go to," because maybe that's something that's untapped. This is another graph of opportunity here. How can we change that?

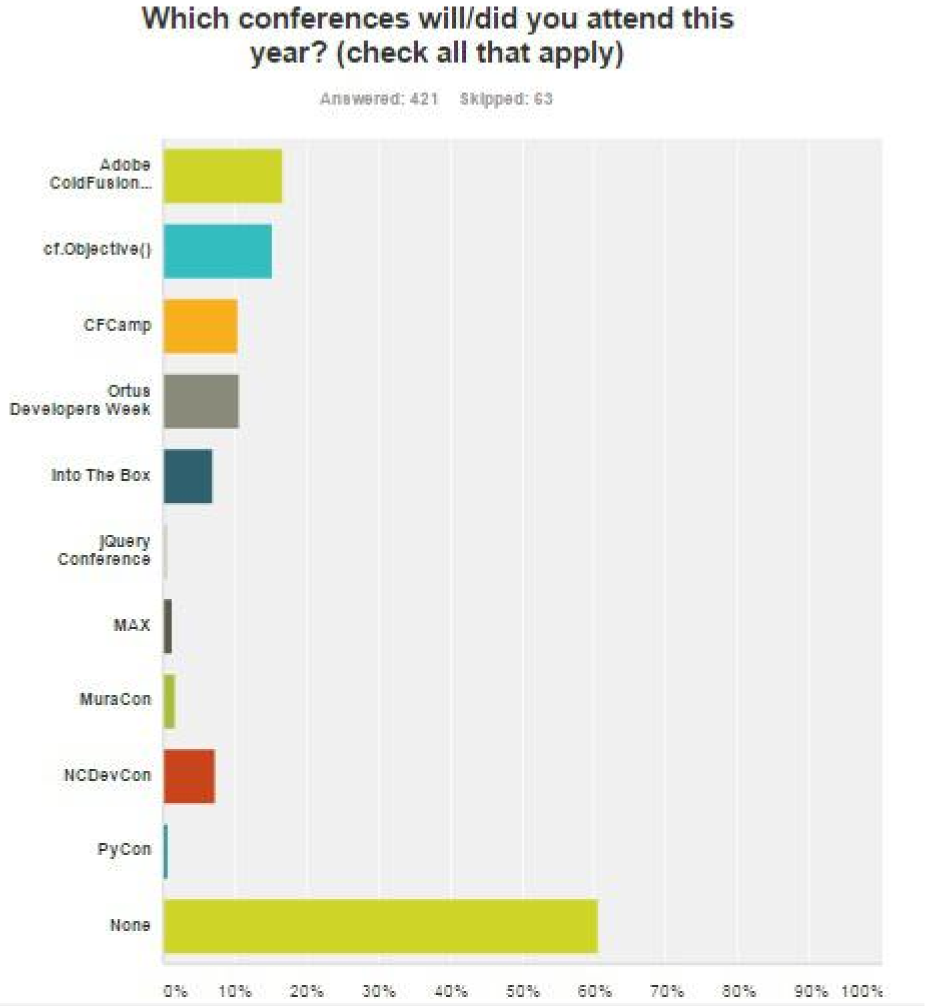
It's going to be way harder for me to talk to a friend and say, "Hey, I code in this language called ColdFusion, you should try it," if I don't have the ability to say, "And why don't you come to my user group meeting this month and take a look at some of the stuff we're working on?" Without that community that’s almost impossible. Communities grow languages, and the ColdFusion community can't grow a language when there's so few attended user groups, in my opinion.

Forums and Slack and what have you are all very great for community, but they are not the same as getting together at a user group meeting or a conference.

Michael’s example:
‘’Or even just explaining a code problem you have. You bring your laptop with you and you get together over a beer or a coffee or whatever and say, "Hey, can you help me out on this piece of code?" I've seen people do that at user group meetings. Definitely. Speaking of conferences, all right, most popular conference is none (see next survey question below). But if they are going to a conference, the ColdFusion Summit, cf.objective(), CF Camp, Developer Week, in fact that's next week I think, right?’’

Conclusion: Networking works best in person, grassroots networking.

24. Which conferences will/did you attend this year?



The one thing that's interesting to learn about this is that out of the respondents of the survey, nearly as many people checked cf.objective() as checked Adobe ColdFusion Summit. If you look at the actual attendance numbers of these conferences, Adobe ColdFusion Summit last year was about 550, CF Objective was about 140, CF Camp was probably about 120 to 150, Into the Box was like 60, NC Dev CON I think is right under 300, and these graphs don't necessarily represent that.

That could mean that there's a lot of people that attend Adobe CF Summit but didn't take the survey, because there were two to three times as many people in attendance, yet a marginally larger number of people. A lot of people that took the survey are probably some of the more active types that already hit several conferences to begin with. CF Summit pulls a lot of the dark matter out of the community that you don't see anywhere else.

Is there even “darker dark matter” CFers that don't even go to conferences but they're still coding in ColdFusion?

Or are those the people who left ColdFusion and went to PHP, or another language?

Also, there are people who have just got a 9:00 to 5:00 job. They don't have time to go on Slack or go to forums or anything or go to conferences, they just want to get their job done, but they're still coding away in CF.

A good question is: would you attend a conference if you had the opportunity?

The most probable answer is that a lot of people in the none category who would be like, "Man, it would be awesome to go to a conference. My company's not paying for that." That's something I would like to see that's not represented in the graph here.

CF Camp is a CF conference held in Munich in Germany. It's at the venue right there at the airport. It's a smaller field, but it has a very family feel to it. It has great food, great location, and it has regular sessions just like every other conference. Has a lot of after activities. They do a code war that Gert Franz helps organize every year. Brad had to sumo wrestle Chris Schmitz in a sumo suit the first year he went. They have a lot of fun after-party stuff. At this point, it's really the only CF conference in Europe left. If you're either in Europe or you have an excuse to go there, then that's good to check out.

Filling out CF surveys

Usually, there are more respondents once the results are out.

We are coming back another day and will take a look at some of the other ones because there's definitely some more interesting questions further down in the survey.

When we put out, we say, "Okay, this is the final survey results," another flurry of people say, "Oh yeah, I forgot to do it," and they come in and fill it out.

Even though it's a relatively small sample of the population, State of the CF Union survey is the largest out of the ColdFusion surveys, so probably one does get the best sampling, from the survey.

And it's a free survey. We tell as many people as we can about it and if anyone . There were a few comments from people saying, "How can we get 'regular ColdFusion programmer starts the survey'?"

They're not part of the social spheres, and it's very difficult to get things in front of them. It can be frustrating sometimes. There's people that do show up sometimes at conferences and you ask them about a certain technology and if they're using it. It might be three or four years old, and their response is, "I've never heard of that." That's one of the frustrating parts, this other area of opportunity is increasing the engagement of the CF community out there so people are more familiar with what's going on.

CFML Slack has been a fantastic platform.

Brad’s example:

‘’Recently, somebody asked on CFML Slack about a querying language for ColdFusion, and I happened to see it and so I gave them a link to a query builder project that Eric Peterson made a while back, and like four other people all saw it and said, "Oh, that looks really amazing. I'll try that." I was thinking CF Slack's sort of like the telegraph of our time for ColdFusion developers. It's really brought the communication points together so it's a lot easier for people to hear about stuff like this. The trick is just you have to onboard people to get them there. ‘’

A lot of people in the *dark-matter-part* of the ColdFusion community don't even know these conferences exist. In fact, they may not even know CF version 2016 exists. No one’s trying to make fun of people, it's just how would you hear about it if you're not on a mailing list and you're not in a social community?

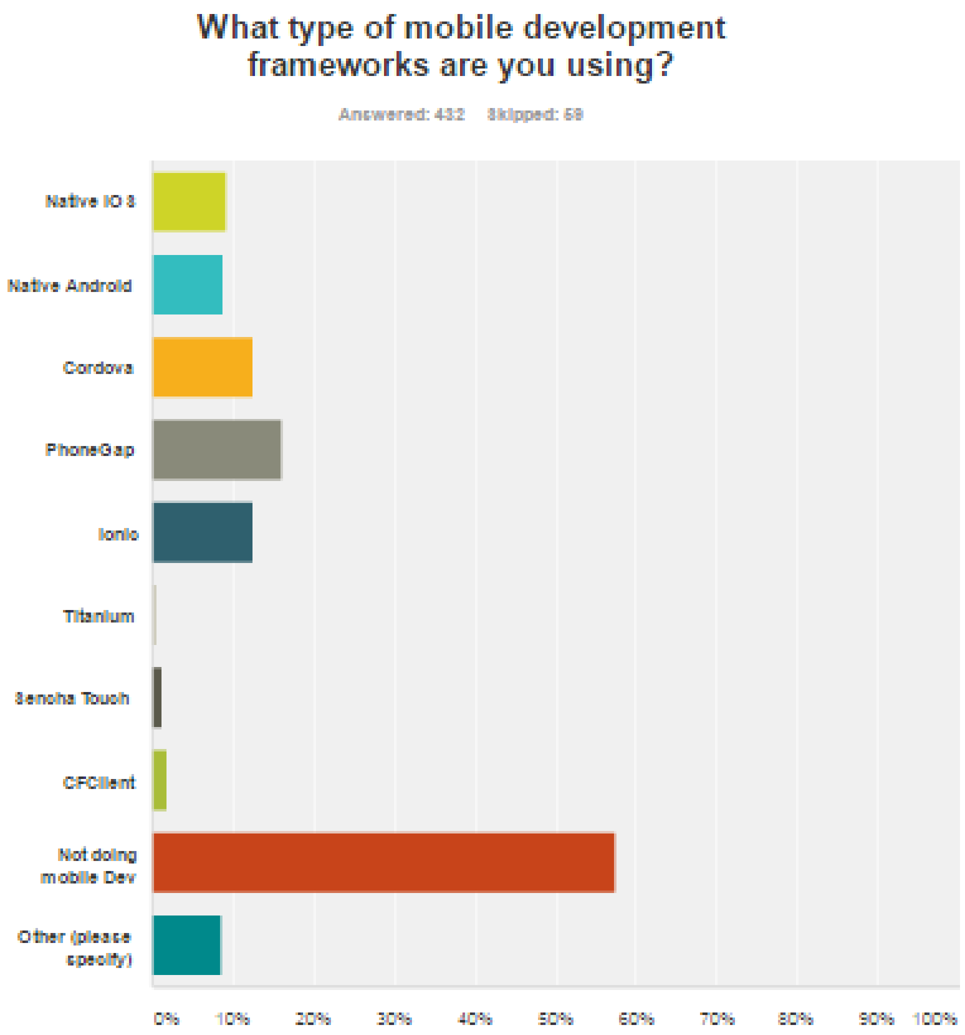
They may not even know that Lucee open-source exists and that there's a free open source version of CFML.

One of the good things about this survey is a fair number of people have said after reading the results, they suddenly realize some technologies that existed that they had never heard of.

There were a couple of write-in responses. Someone said, **"I had no idea there were so many options in ColdFusion."**

This could be a useful thing. It takes a lot of work to put the survey together and it takes a lot of work for everyone who fills out the survey too, but it's very valuable because it does make people aware of what new technologies are out there and what trends are out there.

25. What type of mobile development frameworks are you using?



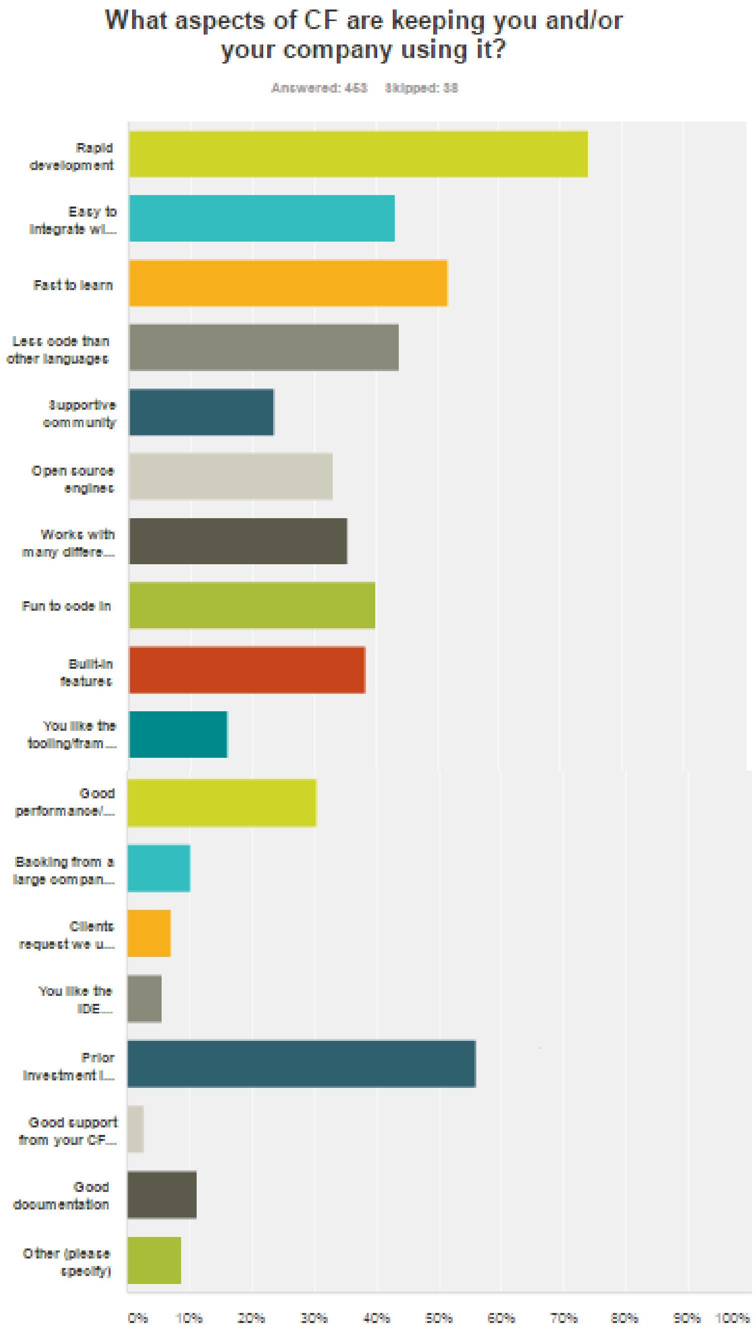
This part is surprising. I thought more people would be doing mobile development by now.

It is surprising to see just the amount of people doing Native IOS and Native android. Brad thinks this may be just a personal thing and he said that he would never really consider that in favor of some of the cross platform solutions like PhoneGap or pretty much any other solution that's not Native. We don't think that a lot of companies are willing to give the giant proverbial middle finger to all android users and say we're only going to support iOS. Because the same number of people are pretty much doing Native IOS as Native android. This suggests that they're doing both.

Brad points out in the responses, that it looks like next year we definitely need to add Xamarin as well as React Native. There were a lot of write-ins for both of those. That's good to know. The other group is still only maybe 10% but it looked like a fair number those were using Xamarin that react Native.

The most popular mobile frameworks is PhoneGap. Brad’s personal favorites is CF client. CF client of course is just a wrap around PhoneGap.

26. What aspects of ColdFusion are keeping you or your company using it?



Very interesting is to see the top reason given is rapid development followed by prior investment in ColdFusion which is interesting. CF is fast to learn, easy to integrate, less code.

A lot of people have mentioned that they know plenty of other languages but they can develop apps a lot quicker in ColdFusion.

If an alternative language is tedious to code in, do you really want to be doing it?

I think in reality, that is a large draw especially for people who pick up a language in their free time. If it's not something that's fun to write in they're probably not going to keep doing it.

A fair number of people put good performance and actually that's surprisingly true because in the ColdFusion engine inside the server, there's a lot of tweaking around. Gets multi-threading and other stuff that gets a lot of throughput to go through a site.

The JVM has been well proven as a platform that scales exceptionally well. Twitter had a big write-up and they switched from Ruby on Rails to the JVM in JAVA. They were able to have a lot fewer servers and support a lot of higher traffic. That’s why it’s said that ColdFusion is a JVM based language because JVM is really a good thing as far as deployment and speed. And as many different manufacturers of the JAVA Virtual Machine are out there, so we know it's going to be around for decades.

Brad likes the built-in CF features as well. That's one of the things he tends to talk about when he talks to non ColdFusion programmers, about that fact that CFML is a platform that comes out of the box with just a whole slew of integration and things that are packaged into the language.

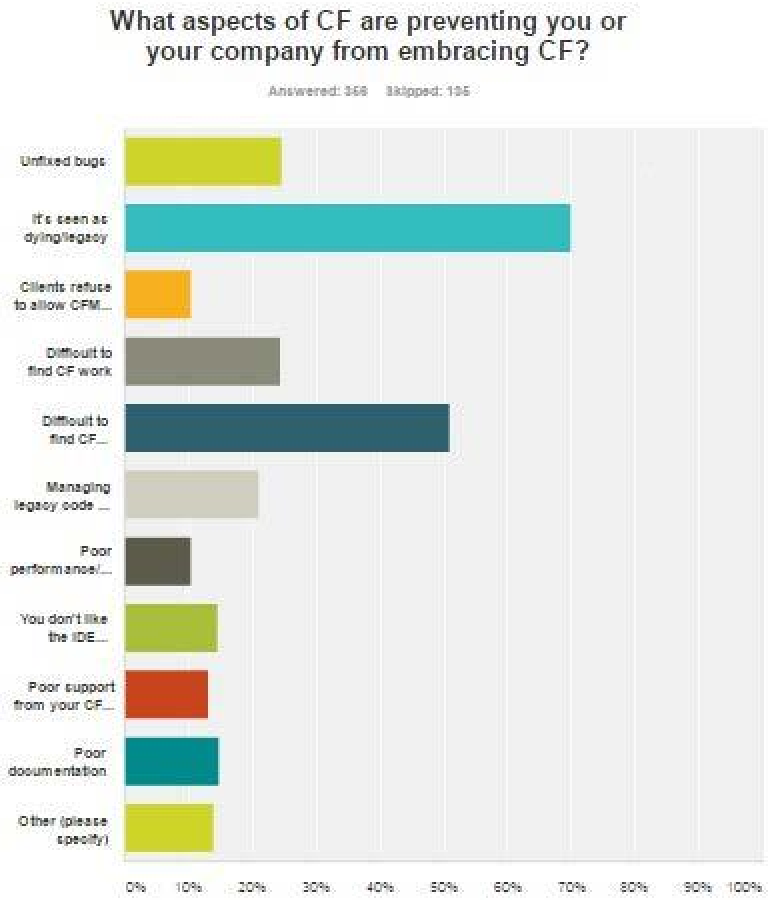
You can write the same thing in PHP or .Net or whatever but you're probably going to end up third party libraries in and if anyone is added a third party library or rolling the dice as to whether it's going to work out immediately or you're going to have to tweak around with it or have compatibility issues with other libraries. It's certainly a big help when things are built in and pre-tested for compatibility.

And, ColdFusion incorporates a large number of open source JAVA libraries but they've gone through the effort to make sure they're all going to work cohesively together. You don't have to worry about that, which is a good thing. It works with many different databases and possibly this one is which some people need to move around between databases and that's certainly a good thing. Then also you can run it open source. Using Lucee open source CFML. In fact you can set up a whole deployment stack that's completely open source using ColdFusion.

There’s a good number (33%) of respondents saying that open source engines are important to them. Mostly Lucee and to a much less extent Railo, and into a incredibly small percent Blue Dragon.

There are 12 or more really good reasons to keep on using ColdFusion. Great to see that and thanks for the suggestion that someone made last year to have this question in the survey.

27. What aspects of ColdFusion are preventing you or your company from embracing?



So the biggest one by far is seeing as dying in legacy which is sad because that's really completely a perception. That's not a fault of the languages self or what you can build with it, it's really just a giant perception issue. That's what drives some CIOs and CTOs types away from CF. You'll come in and think, "Let's use .Net." Nobody got fired for buying Microsoft as the saying goes. That's one of the biggest issues that we need to try to fix as a community. It’s just the perception of ColdFusion rather that the reality on the street.

As the answer says, it's seen as dying legacy, it's not that it actually has died. So, what ‘legacy’ means exactly? Whether it just means the code being around for a while and may be poorly written. But you can see some really modern and exiting things happening in ColdFusion code these days with Docker deployments, micro-service architecture, REST APIs and Command Line CF for automatic builds and

Continuous Integration.. It's cutting edge but we need to get the word out there that yes, it is alive and it can be modern code. Sure you can write old style code in it and you may be maintaining mountains of old style code if you're “lucky”.

Many ColdFusion developers aren't aware of the modern tooling and resources available. Imagine that issue amplified even more in the communities of people who aren't even doing ColdFusion.

Here's the irony. Here’s actually a response, here’s a write-in response in this question, lack of command lines sucks. Command box is precisely that. A command line tool for ColdFusion has been around for three years now and here's a member of the ColdFusion community who literally is three years or more behind. We've had a command line CF interpreter for three years and they apparently don't even know it.

BTW, for if you don't know the cost of CommandBox, it is absolutely free. (It is open source).

There should be no reason to not be using that so you can get the benefits of spinning up servers and different versions of ColdFusion and all the other command line things you can do. The people from the Lucee initiative are working in the next dot release speeding up how quickly it loads. When you use a command line, it can do it twice as fast. When you run a command line there's a little bit of loading in the background to do this.

The next biggest issue is difficulty of finding CF developers. It's interesting because we have both sides of that coin. Some developers have difficulty finding CF work and some companies are finding it difficult to hire CF developers. Both of those have a decent amount of votes but it looks like there's more people looking for developers having trouble. That's good for developers, that means that's a hiring market.

Here is a radical solution that works for us: stop looking for CF developers! Stop limiting your job search to only applicants who already know ColdFusion but also look for qualified developers that know a similar scripting language such as Ruby, JAVA, .Net, PHP. And simply train them on the specifics of ColdFusion. It only a takes a sharp developer a week or two to pick up the syntax of CFML because the basic ideas of coding are the same as in other languages. They'll be able to fill a lot more seats that way.

Also don't limit your search to people who happened to live in the same town that you're based in, because there's a lot of remote availability.

Unfixed bugs in the CF engine is the next biggest one which is kind of interesting. Yes, it can be personally frustrating when you've submitted a bug, maybe two or three years ago and it's still sitting around a bug tracker. There's two problems for that. We don't know if the people who answered that were referring to the Adobe bug base or the Lucee bug base. Obviously there's nothing that we can do to fix Adobe bugs other than voting and commenting and pointing them out. It's worth pointing out that if there are unfixed bugs in the Lucee CFML server at least that's something the community has the ability to fix on their own terms to be able to submit pool request and things. At least there’s reprieve available for people using the open source engines they have the ability to help contribute in that area by directly fixing the code in question.

And lobby directly to the people who might be able to help. If you have a relationship with some of the people who do the committing to that code base you can convince them why it's important to fix this particular bug. You can also help with debugging and documenting use cases that demonstrate the bug it to make it easy for the fix to happen.

It is worth talking about how people can push for bugs to be fixed. For Lucee specifically, the product manager now is Patrick Quinn of WebApper. He's been pushing for monthly sprints inside of Lucee development and he started a thread on our Discourse forum which by the way is where we just moved our Google groups to. If there is a bug that you want to see fixed he does look to see what votes are on bugs to priorities the fixes. But also pointing that out in the discourse forum and tagging him is the way to get something flagged for Lucee.

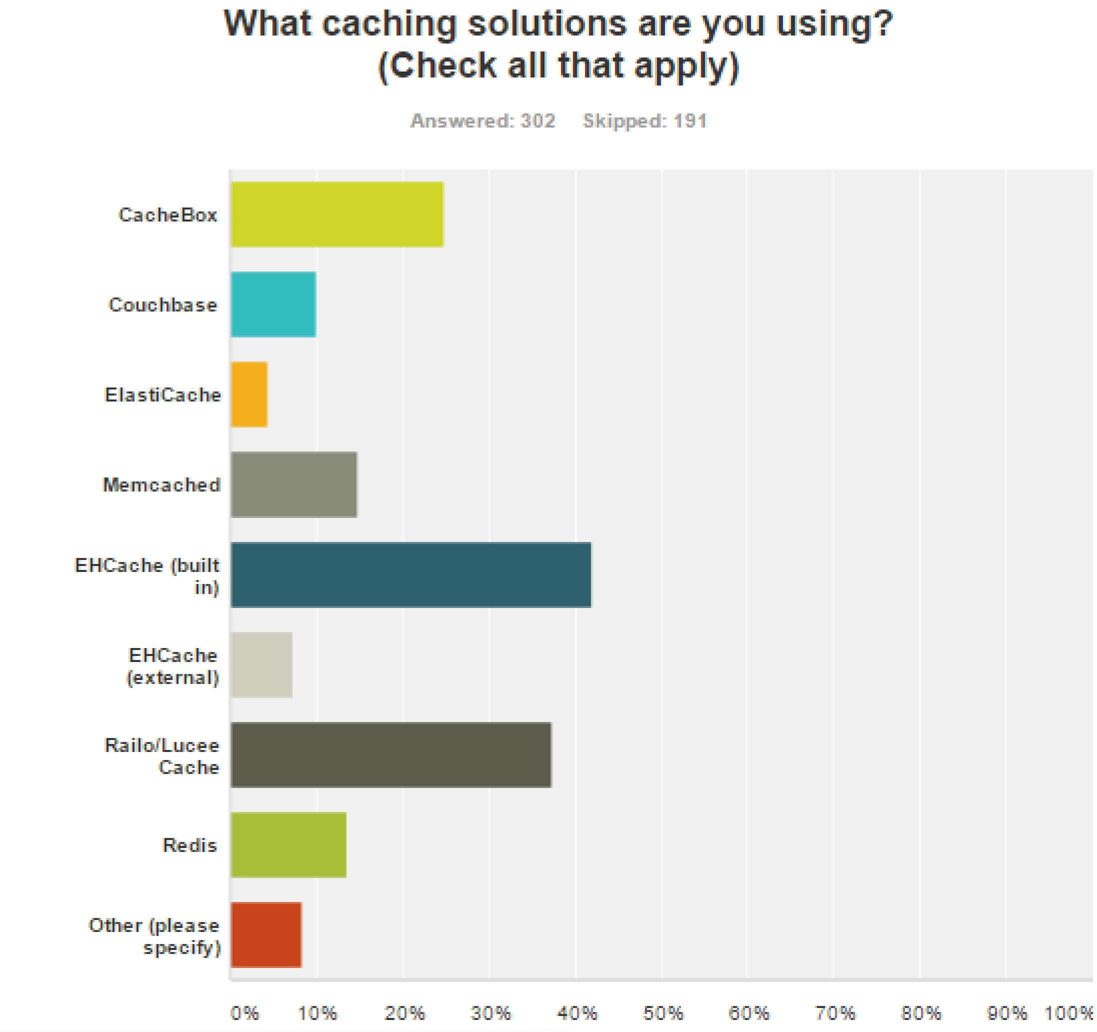
Just to encourage people there, it helps if you explain why this would benefit other people and not just yourself, why it's an important bug to fix. The other thing is if you have other friends who are having issues with the same bug, get them to comment too, do some little grassroots outreaching, get some encouragement. If you have several people saying, "Let's fix this." It will happen sooner.

Every once in a while you'll see somebody say, "Why has this bug not been fixed for either Adobe or Lucee?" I'll look and say, "There's zero tickets ... There's zero votes on the bug." For starters, find some people to vote on it. The equivalent for Adobe, they do have voting on their bug tracker. A lot of people come on and we'll go tag him and say, "Can you follow up and see if there's a status on this bug." That's a good way if you have a ticket and the Adobe bug-based to kind of get a push in that area. Obviously they can't fix everything all the time but that's kind of two good things that people can follow if they have bugs in either engine if they'd like to lite a fire under.

Brad said that he appreciates both Adobe and Lucee for having public bug databases because Lucee has always had one as did Railo. But Adobe didn't use to have this and they started about 10 years after TeraTech created the CFBugHunt.org public bug list site for CFMX (CF version 6).

If you remember ColdFusion 6 you'll also remember that ColdFusion 5 was incredibly reliable. Damon Cooper of Macromedia, had a thorough QA process. It would recreate the code to demonstrate a bug existed. Whenever someone reported a bug they asked him for a copy of their code and then they would stick that in their code repository so that a new version of ColdFusion would get run through the same code that had bugs before so they wouldn't come back as zombie bugs.

28. What caching solutions are CFers using?



Number one caching solution is EHCache. Of course, that comes built-in.

It’s important to note that Adobe ColdFusion has EHCache built-in for like the CF cache tags, the cache get, cache put functions. Lucee and Railo have a bit more built-on to that, they kind of have a caching mechanism being created multiple named cashes that point to some sort of internal or external data store. Railo and Lucee offer to you EH cache as well as Adobe ColdFusion built-in, and they also have a RAM cache option which stores it memory. It’s not surprising to see that both of those options are sort of by and enlarge the most commonly used since they require zero additional effort, they just come out of the box.

Lucee server has built-in EHCache in the same manner the Adobe ColdFusion does, out of the box.

And there is CacheBox of course. It’s mainly an aggregator that allows you to connect to external cache stores in a similar fashion that Railo and Lucee do. And it supports an in-memory cache. People using CacheBox could quite possibly be connecting to Memcached or to Couchbase or to a JDBC store. Obviously, we can't tell from the raw survey data, but it’s interesting to see the breakdown here.

For people who aren’t doing caching (which seems to be about half of the people who took this survey) what kind of things would you want to cache in your application and what kind of effects might you expect to see?

Caching is kind of one of the go-to patterns and strategies when you are dealing with high availability and you are trying to work a performance especially with some sort of distributive systems. A couple of quick and easy examples we use caching on the order solutions blog for instance to cache both user sessions. If you are visiting a site logged in or not as well as HTML content of the blog post. It’s things that change very infrequently but is accessed often. Typical caching strategy are used on servers is to have Couchbase cluster sitting behind CacheBox so we have just Lucee’s cache aggregator and we’ll be caching the HTML of the blog post inside of that.

Having an external cache can be nice. External meaning not in your server’s RAM which would be an in-process cache. Having an auto process cache is nice because if you restart a server you don’t have to wait for those cashes to warm up again. We can bounce the server when Lucee comes back up or ColdFusion comes back up, it’s still pulling hits in the cache mechanisms because that stored out of process.

That’s some really nice built-in functionality. Adobe ColdFusion will only let you cache queries in the built-in EH cash. Lucee, Railo will let you point your query cache to any external cache you want so you can cache queries in Couchbase or Memcached.

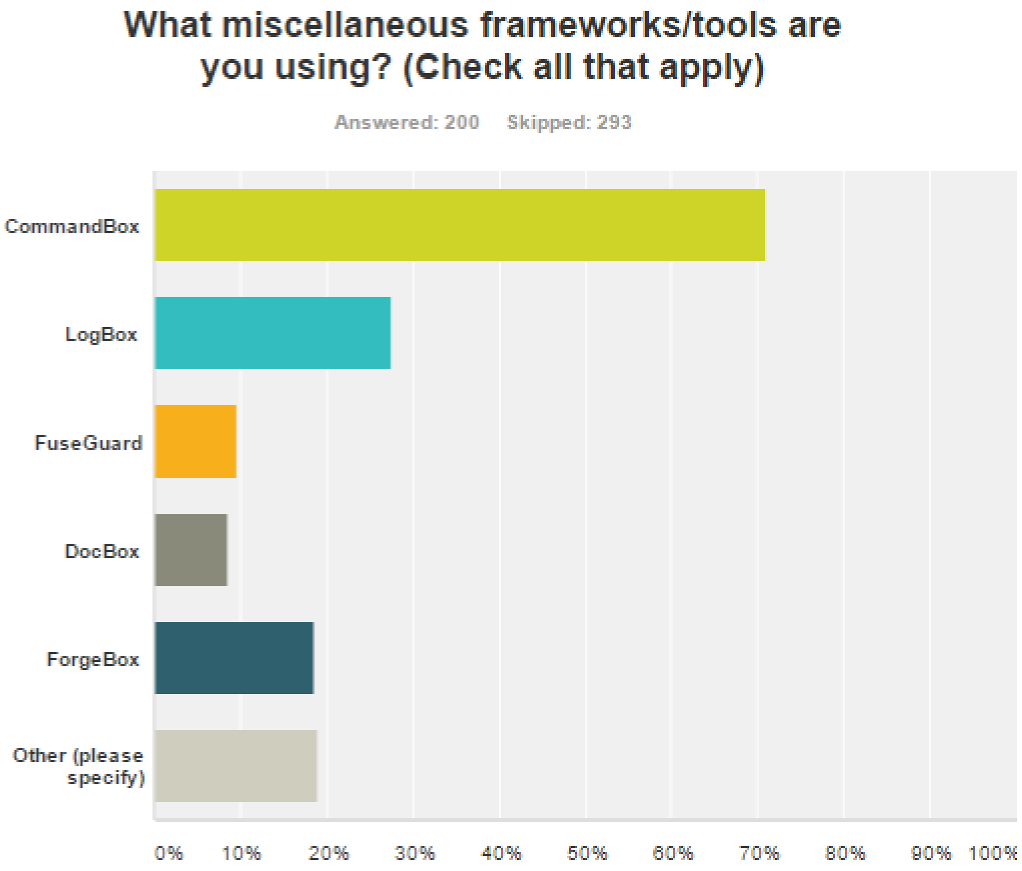
Couchbase as a caching solution is a distributed database. It is sort of a marriage between Memcached and couch DB.

They eventually had children. It was called Membased and they renamed it the Couchbase. It combines both the no sequel data sort including an actual sequel-like querying language so you can cache documents but it also works just as a typical key value cache. You can throw binary blobs in it - so you can cache HTML.

But it’s a little bit like a CDN, it’s spread out over several servers probably geographically spread out. You can have as many nodes in the Couchbase cluster, you can have cross data replication. For those of you who want to speed up their apps start off by looking at the sequel as Bill Clinton might have said, “Is the sequel stupid?”

Brad also pointed out that CacheBox does end in Box but that’s not a requirement to use ColdBox with it. CacheBox is something that comes along with ColdBox but it’s a standalone library. If you have a legacy application or a FuseBox or Framework 1 app and you want to drop in some caching, you can always drop in CacheBox all by itself.

29. What miscellaneous frameworks/tools are you using?



The number one miscellaneous framework is CommandBox which is completely free. Brad Wood is the lead developer of CommandBox. It is a CI tool so it's a rebel, lets you run ColdFusion directly from the command line, you can execute CFM files from a Cron job or as shebang script. But also there's package management you can install libraries, you can install frameworks and you can most importantly start servers. It can start Lucee servers, it can start any version of Adobe servers. You can have four different server versions, Adobe, ColdFusion 11, Lucee 5, Lucee 4

This works because they secure the port so nothing conflicts. You can use CommandBox in Travesty I to test frameworks like ColdBox against Lucee 4 or Lucee 5, ColdFusion 10 and ColdFusion 11, ColdFusion 2016. This is all based on CommandBox. You could say start the server and it boots it up. The CommandBox use jumped a lot when it began starting Adobe servers.

Then LogBox, the second most used one by 27% of folks, why would someone want to use that over just using CF log? Like CacheBox, LogBox is something that is a part of the ColdBox family but it's a standalone framework so you can use it in any application you want even if you are not using ColdBox. But LogBox lets you do similar to CF log but just much more flexible. You can have a generic concept of, "I have a log message of this severity," and it can go to a text file, it can go to a database, it can go to an email, it can even go to a Tweeter feed. You can dynamically at provision where you want these log messages to be logged to and that can be different for different environments. On Dev, you can say, dump all log messages of any kind into a text file that gets zipped up every 10 megabytes, then production may be you email the issue. It can be very handy especially if you are dealing with any kind of containerized deployment of your apps.

For example if you are using Docker containers to run you apps LogBox is very useful. Because there's no static files system underneath a container, so it's important to be able to log your error messages outside of the container. We have some custom appenders written for things like bug log HQ so we can have our servers all via LogBox logging in the error messages on the site to some centralized bug log HQ server. It comes in very handy for situations like that.

It would be good if more people were using FuseGuard. This is a product of Pete Freitag from Foundeo and it's excellent. It's basically a web application firewall that's written in ColdFusion and it will actually block a large number of known hack attempts from being able to hit your website. If someone tries to exploit a vulnerable SQL statement or SQL injection or exploit a path transversal exploits, then FuseGuard will detect those via some configurable heuristics and it will block these requests and even log them. It is a commercial product but it's very well worth the cost.

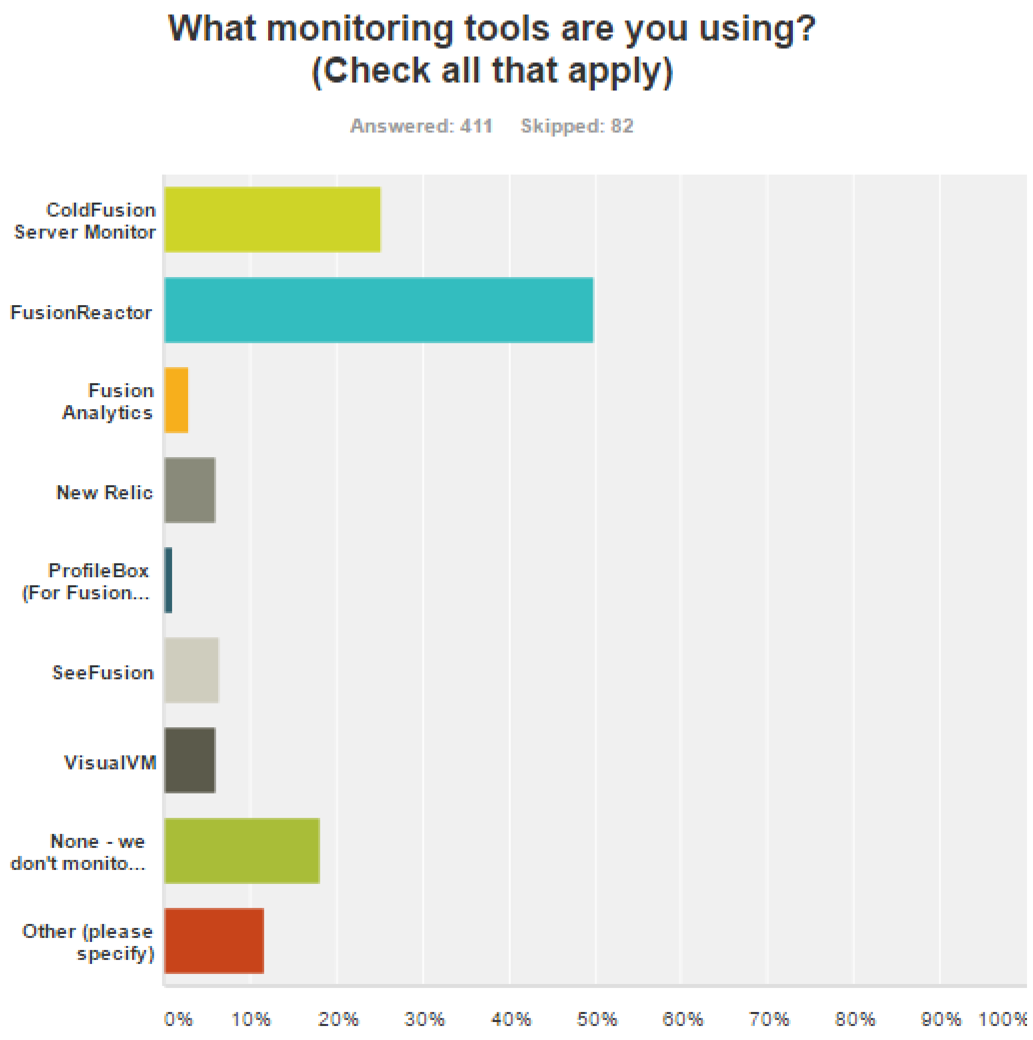
It's a fantastic product, Pete's really done a good job with that one so I think more people should look into it. On that note if you're curious about your servers, Pete the author of FuseGuard also has a HackMyCF.com website which is a very cheap subscription you can pay and it'll actually email you and tell you if you have any vulnerabilities on your website. It's a non-intrusive scanning that it does and it will alert preemptively, "You are behind in your updates, you need to lock down these directories." It may scare you the first time you run it when you discover how many server security issues you have.

That's a great tool and it will alert you if there's a patch being put out by Adobe. Just to be clear, HackMyCF is just looking at the server side of things. It isn't checking code for security issues. FuseGuard is protects some holes you might have in your code. We have a question a little bit further down where people admit to how much they're been hacked so we'll look at that a bit later. Then two other tools here miscellaneous frameworks, DotBox and ForgeBox.

Very quickly, DotBox is the oldest folk of the ColdDock project which I believe is Mark Mendel. It was no longer being maintained so we kind of took it under our wing and rebranded it. It allows you to generate HTML documentation of your CFCs. We use that for the ColdBox, TextBox, all of our internal libraries, we have API docs you can use, we use DockBox to generate those.

ForgeBox is less of a tool and it's actually a website. The forgebox.io website is where we host packages that you can install from CommandBoxes, kind of part of our ecosystem. But it's not a requirement that it's ColdBox-specific. Create an FPR, whatever you want you can put them on ForgeBox.

30. What monitoring tools are you using?



The number one tool is FusionReactor which maybe is no surprise because it's the one that gets the best maintenance and new features added to it the fastest. They have releases nearly every month. They're also very supportive the ColdFusion community. Their sponsor CF Camp, NCDevCon and Into The Box among other CF conferences. They put a lot of effort into reaching out the ColdFusion community.

It's worth noting SeeFusion has not very many votes. It's a product that has gone stale for a while. It is created by WebApper. It used to be a great product and still is okay but it just hasn't been getting any updates. Although they have recently released SeeFusion 5 which involves a lot of cloud based management and pricing. It will be interesting to see if SeeFusion picks up any usage just based on the fact that they are starting to push out a new version.

There is a Podcast interview with David Tattersall who is the CEO of Integral and apparently they’ve got a cloud based version of FusionReactor too. It'll push the metrics from your websites to a cloud location where you can view them externally which is cool. They’ve also got lots of other useful features like, they can automatically restart the server, they can tell you when it’s getting close to crashing, memory limit or disk-limit or whatever limits you put in. It also can help you debug if you got something slow or crashing it can help you identify the slow SQL or the slow ColdFusion files.

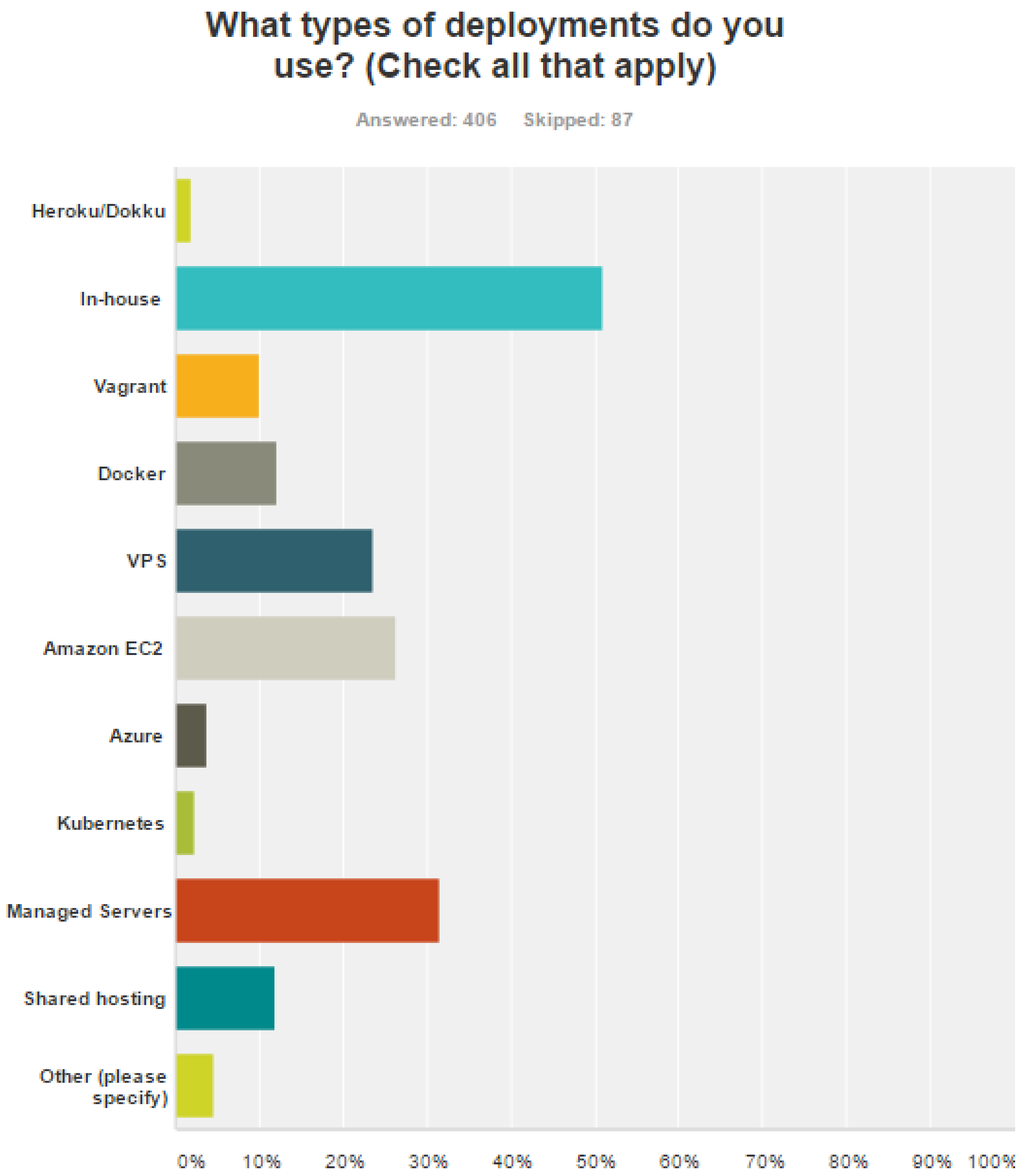
It is slightly disappointing to see almost 20% of people saying they don’t monitor their servers at all. Does it mean that they don’t personally monitor it and some of the IT department does?

A lot of people will come on the mailing list or such and they'll say I've got a page that takes 10 minutes to run and I don’t know why or crashes my server. I will ask if you've got a FusionReactor or SeeFusion or some other CF monitoring tool installed; and the answer is no. The fact of the matter is that they’ve wasted far more of their company’s resources and their time than it would have taken just to purchase a product like this. If you’ve got ColdFusion enterprise you already have a monitor that's not nearly as well featured as FusionReactor but it'll at least give things like stack traces and memory usage out of the box. I think a lot of people just don’t realize how important monitoring is. Brad recommends that everybody should have some monitoring solution under their belt that they can use to see what their code is doing before a server crash or slow down happens.

Now the FusionReactor has a monthly subscription pricing as well as just being able to purchase the software. Their lowest pricing is $39 a month which is cheaper than getting a cup of coffee every day at a little over one dollar a day.

FusionReactor also has developer license if you're going to only use it for your developer work station not on a production web server and it is around $200 a year which is much cheaper than their production pricing of around $1500. So there's a lot of options and they're well worth the time you're going to spend.

31. What types of deployments do you use?



Lots of tools people could use on deployments, most popular one is deploying in-house and second one on managed servers. Those are where the deployment is going to which is slightly different question but this is the multi choice question. Shared hosting was behind on that, because you're sharing a server with other websites and if you used to get the security locked down to where it's nearly unusable, or the security isn't locked down which means that you're not very well separated from the other websites on the same server. To put this in a metaphor, shared hosting is like you are going to a hostel dormitory and you with all the other people and if someone gets drunk and throws up on the bunk bed you have a problem. A managed server is like an apartment complex where there are other apartments but they are walled off from each other.

With a virtual private server (VPS) the ISP manages the hardware but you install everything you want including ColdFusion on your own (virtual) server.

A managed server/ dedicated machine is like having a detached house and you have to furnish it and fix the facet when it leaks. The ISP builds it for you and it's totally separate from anyone else.

The number one software deployment solution is Docker.

Docker isn't necessarily mutually exclusive, there are a number of these deployment software solutions. For instance, Kubernetes is an orchestration layer on top of Docker. Heroku and Dokku are sort of layers on top of Docker as well. Docker is very attractive and what I see right now in the cloud world is that everything is trending towards Docker in some form or another. It's the idea that you can completely separate the server that you're installing your code on to from the application itself so you have this whole precious snowflake of an application that you bundle up in this nice little package and you decide to deploy 10 of these. The orchestration layer on top of Docker lets you configure that if the load of my application reaches a certain threshold, deploy two more servers and automatic load balancing between them.

That gives a level of flexibility that's unheard of in the in-house category. In Brad's opinion, this question right here is where the ColdFusion space has some of the most to grow in the future. ColdFusion is really lagging behind in this area of deployments. If you were to look at the same question with another major language like Ruby or Node then they much less favor in-house servers. Instead they would be doing Heroku, or Docker based deployment with Kubernetes.

In some of ColdFusion shops like our government, they may have very tight restrictions on where they are allowed to run code. So quite possibly Docker and other modern deployment solutions on the public cloud aren't options for them, based on the red tape and national security concerns.

There's a lot of this Kubernetes, and it is available just a host of solution Google cloud services. You can also run your own Kubernetes clusters effectively in-house. Which gets around that national security concern I mentioned earlier,

Vagrant is similar idea to Docker and is pretty popular. Vagrant can have provisions that point to any cloud service like Rackspace or other cloud servers. But 99% of people using Vagrant and are using it specifically for local development. Vagrant is for provisioning virtual machines (VMs). Docker is for provisioning containers. A container is much more lightweight than an entire virtual machine is faster to provision. Typing Vagrant up from scratch might take 15 minutes, bringing up a Docker instance might take 30 seconds. Docker is the next logical evolution of deployment solutions from Vagrant. In fact when you install Docker today on a Windows machine, but actually install VirtualBox and deploys a single virtual machine too that then the Docker container uses to run your ColdFusion inside it.

So a virtual machine like Vagrant creates for you by default on a tool like VirtualBox is a full installation of an operating system that's booted up from scratch and virtual hardware. You assign RAM to it, you assign CPUs to it, you boot all the way from virtualized bios up to a Linux installation. Docker when we say process is literally just a process running inside of Linux operating system or in the latest versions of Windows 10 support containers natively. When you spin up a new Docker container you just have a base Linux operating system running, and it's literally just a matter of saying spawn this new process. Inside of that is little capsulated Linux thing that's running and it thinks it's its own machine but is really just one of many processes running on a base hardware.

With Docker your density is much higher, the number of containers you can spin up on given hardware is more. The time to spin them up is just a matter of seconds because you're not booting the operating system from scratch. You are just saying here's an existing running OS spin me off a new process. It's a very technical answer but there are a lot of benefits to the container based in that it's just a process running on a base operating system. It's still very segregated system resource wise, but it's a lot more light weight and faster than a full VM like Vagrant gives you.

It could keep your applications separate from each other if you've got a development server you could have different containers that have different versions of ColdFusion or different databases of whatever you were testing.

In fact we have CommandBox based containers that you can just say, "I want to run Adobe ColdFusion 11." That's what it spins up. The idea with a container based approach is you have a container for each application and each version of CF that you are testing it on. They are little separate deployable apps spaces.

The other thing is that once you've configured the container you can save it off to disk and then when you want to bring it back to life you can just spin it up very quickly.

Docker is growing in the future and things like Kubernetes and EC2 are also expected to be growing. Most people if you said how long would it take to put another dedicated web server online starting today, they might tell you two weeks. Anybody using Kubernetes who say how long would it take you to put another website online and they'd be like, I just did it while we were talking! That is a totally different world for deployment.

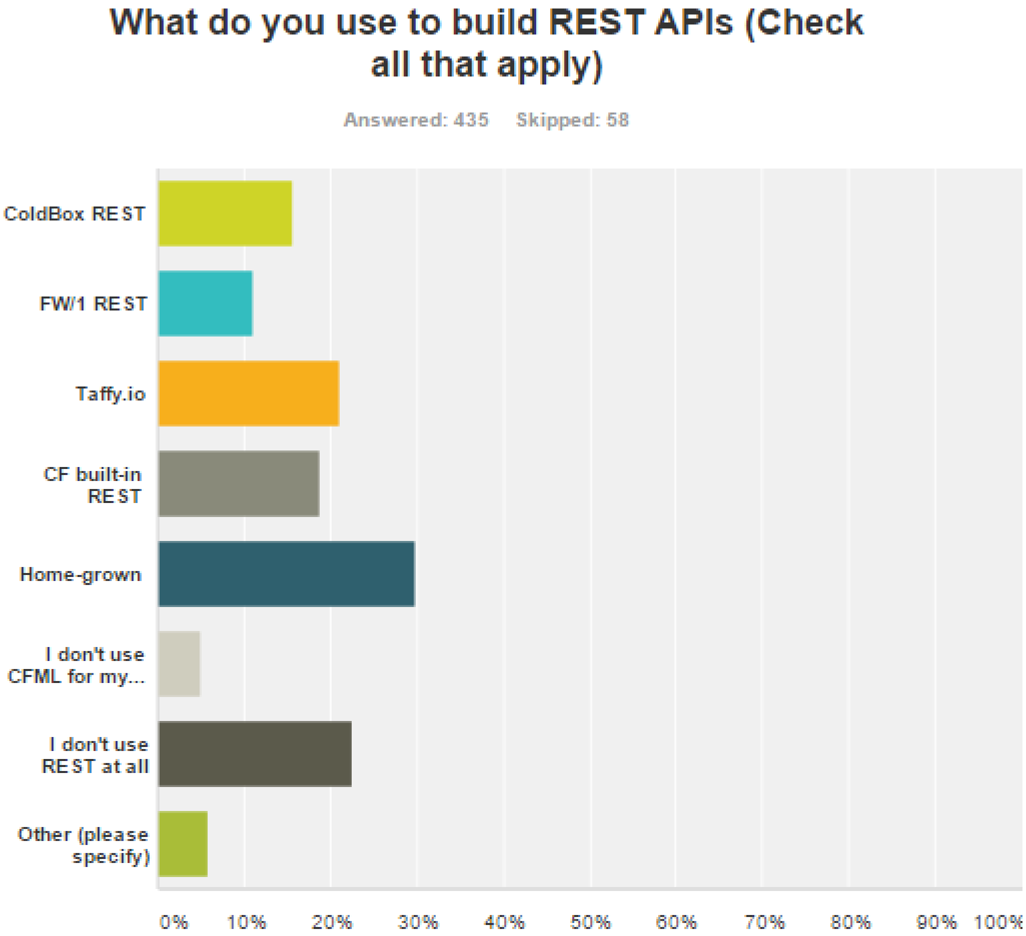
EC2 is the only cloud deployment option that's really natively supported by Adobe right now. They've promised us Docker containers but we still haven't seen it. EC2 is one of the only deployment options that really gives Adobe the ability to collect the pricing, you pay per hour for each ColdFusion EC2 license instance. The other deployment solutions don't quite have the same control as EC2 does to be able to charge the customer per hour per instance.

If you wanted to use Amazon you also could run Lucee.

WebApp worked on making gold certified AMIs for EC2. Lucee also has Heroku built packs, they have based Docker images you can build on obviously with the licensing out of the picture, you don’t have to worry at all about your deployment licensing with Lucee as you do Adobe ColdFusion, because it is open source and free to license.

We understand that Adobe is working on some better cloud-based licensing.

32. What do you use to build REST APIs?



A lot of different things here, the number one response is Homegrown and number two is we don't use REST.

It's surprising that there are a descent number of people not using REST but it is slightly disappointing to see how many people have grown their own. All the major frameworks all have a REST implementation, and so it's disappointing to think of the hours people have spent writing their own. Any of the off the shelf frameworks would save them a large amount of time. Even the CF built in REST which is generally regarded as not being that good at least has descent usage but is not as many as the homegrown ones.

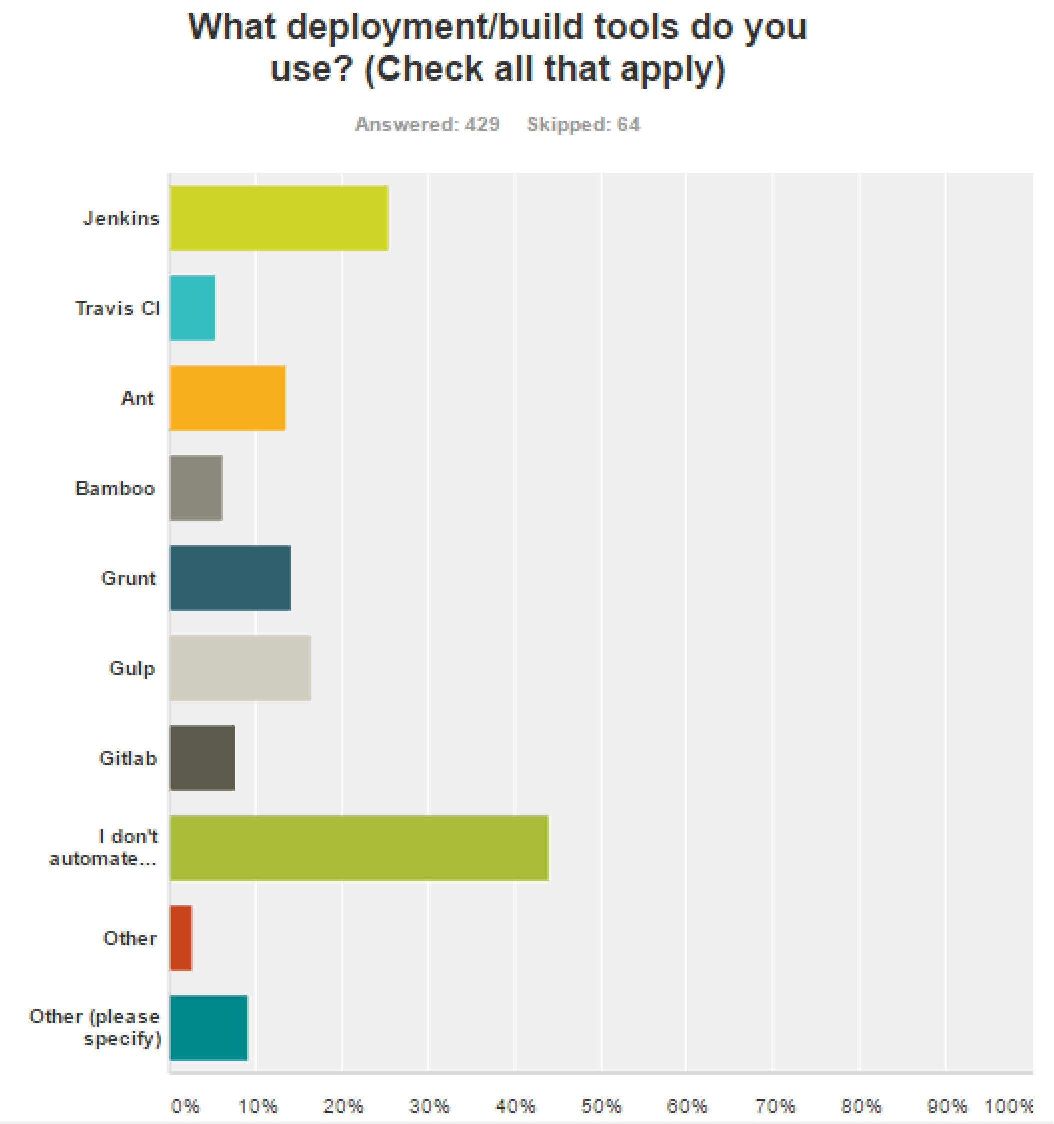
Just for those who aren't using REST, why would you want to use or expose a REST API in your app? What the benefit of that? It used to be the biggest reason people would write a REST API, is when they wanted to integrate with a third party, some external company that wanted to be able to hit their services. Nowadays with the Angular and VUE.JS "single page apps" your Front End code needs a REST

Back End just run. More and more commonly people use a REST API just for their own website. They built ColdFusion based REST Back End that serves up in a stateless manner than they have JavaScript front end that's consuming that. You have a mobile app just using the same sort of data; you can back it with one REST API for both.

It's a way to isolate different parts of code from each other so you could swap out the front end and keep the back end the same or as you said talk to third parties.

But it looks from the write ins responses that even CF wheels and Preside CMS and Far Cry all have some little custom REST stuff on there as well. Pretty much everybody is using some kind of REST tooling and chances are it does a lot more than something you can write on your own.

33. What deployment/build tools do you use?



Most people using Jenkins are probably using ANT inside of Jenkins for Travis. What's our percentage not using is about 45%, we use Jenkins for years as just a general build tool for all of our Box products. When I make a commit to CommandBox or the Jenkins build that kicks off and builds all the binaries and runs all the test but lately we've been investing a lot in the Travis CI (Continuous Integration) and it has a very tight integration with GitHub. One of the best things at Travis if you're an open source author is when people send your product pull request as soon as they send the pull a Travis CI build, which by the way is Docker based, it’s a container that spins off the clouds somewhere.

It will spin up a container somewhere in the ether, it'll run your entire unit test against the pull request that’s being submitted to you and it will tell you right there in GitHub, test to pass or test to fail which is great because it gives you instant feedback. As an open source author this is great. Someone sends you a pull request and you can tell them right away well you broke the build come back and talk to you once the tests are passing which helps keep code quality high.

The 44% of survey respondents who aren't using build tools are missing out on getting a reliable build made automatically where every time you make a change in your code it can run unit tests, it can deploy it's different servers depending on how you got the things setup. And we believe among CF developers who didn't take the survey the percentage is probably even a larger. What a waste of time and money!

Can you create build of your application with a single click? Chances are if you are not using one of these build tools, you'd probably have to answer No to that.

The reason you wouldn't want to do this manually is it's very easy to forget to copy a file or to mis-copy things or get settings wrong and then you have bugs to track down that could be very hard to find.

A few years ago every time a new version of ColdBox was released, Louise would build it manually on his computer and test it. A lot of companies are the same way - you have one guy who knows all the stuff that has to happen to get the app into the production. Anybody else needs to do a build they're screwed and so that's why we started using Jenkins internally. So anybody can build the app.

Jenkins, which is a fork of Hudson, is free. Travis CI is also free as long as your projects are open source. ANT is also free - it's an Apache product. ANT is more the xml files you use to describe what you want your build to do. ANT is typically used inside of Travis or inside of Jenkins. Bamboo is the equivalent to Jenkins ties in with the tool set.

A lot of these options are open source or free.

There shouldn't be a pricing barrier for sure. They are all Node based. We actually use those specifically Grant for a lot of out front end builds. We have a Lexar project now with ColdBox that deals with things like your style sheet, compilations, your job or script all that asset dealings that you do during a build and so we use a lot or Grant and Gold and of course there's no cost for that, those are very popular, there's plug-ins for everything under the sun.

GitLab is like a self hosted GitHub repository that you can have behind a fire wall in your own appliance and it has its own Jenkins automation built in. Jenkins lets you manually trigger builds or even schedule them. GitLab actions are mostly based on a commit just came in. It's all based in the concept, you have repository somebody commits something to it when you run a pipeline of build tasks from it.

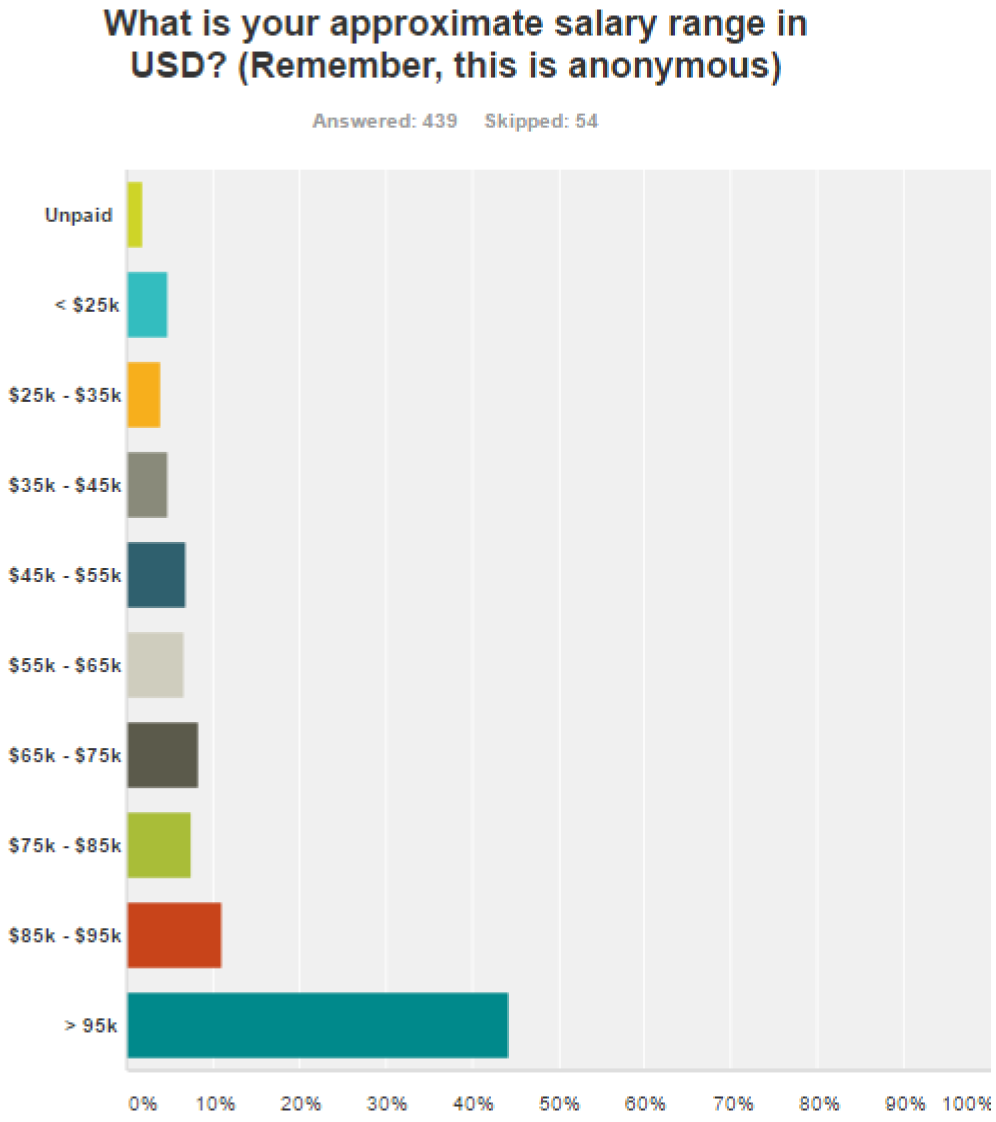
I recorded an in depth interview about GitLib with George Murphy, who is a speaker at Into The Box Conference

Before we move on, related to this is doing continuous integration which a lot of these tools let you do. Why would someone want to do continuous integration?

The idea between continuous integration is all about closing that feedback loop between the amount of time that you write a line of a code and the product is build and it's tested and you know whether or not that works. That can be feedback from a set of unit test, it could be feedback from just a tester on your continuous integration is all about I'm made to commit and lets run the unit test on those, let's deploy that to the stage and environment. Let’s use some JSlint, Variscope or whatever other scanning tools you want to run and it’s automated so it’s not a manual process as opposed to saying I need to push that code to stage and there's just so much work I'll wait till Friday to do it, it should just happen. There's no reason why your code can’t be on your staging and production server within five minutes of you committing your code changes, and all without you lifting a figure.

If the unit test don’t pass it shouldn’t be deployed, the build should be failed and you should be notified. Because why have a human bother testing your code when your unit test have all ready told you that they are failing now. It’s all about automating reducing those feedback cycles, improving code quality.

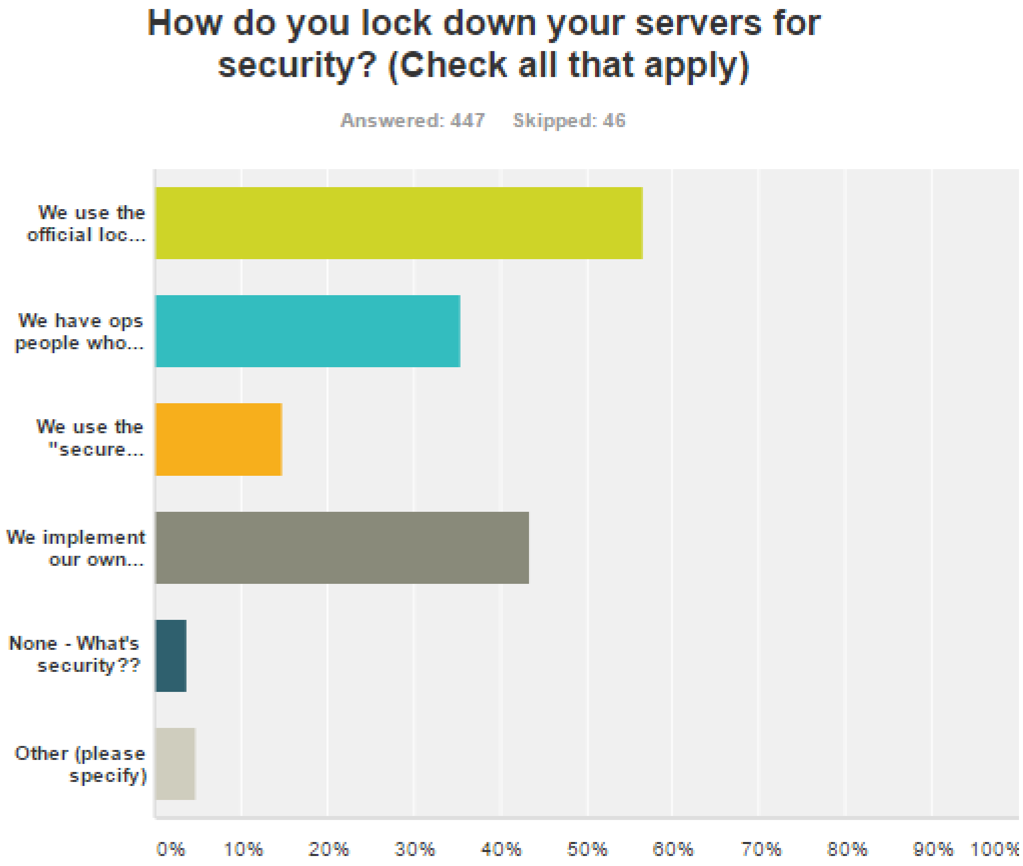
34. What is your approximate salary range in USD?



Nearly half of the people were making more than 95K a year and then there was pretty even distribution all the way down to 25K which is hopefully on an intern hobby level.

There was a couple unpaid but this is great news for ColdFusion developers. If someone is considering should I develop in ColdFusion it goes along with the other question that said it's difficult to find ColdFusion developers, so instead it's a market to be hired in. People are willing to pay to try to get people to come be a ColdFusion developer and this is good news for people with ColdFusion skills. Next year we will include higher salary bands to get some better data at the top end.

35. How do you lock down your servers for security?



We need to shoot the 17 people that said “None – what’s security??”

The people who took this survey tend to be the most clued in ColdFusion developers who are on the forums, who are on the slack channel. The people who just do a regular nine to five job I guarantee that way more than 3.8% don’t do anything for security.

The official lockdown guide is great. Pete Freiteg again helps write those, he's the author of FuseGuard and HackmyCF.com. A lot of people just don’t even know those exist. They are freely available PDFs that literary gives you screenshot step by step guide to improving your server security. From Apache to AIS you name it everything you need to do to lock down. If everybody followed the guide and updated their server with the recent updates, the majority of ColdFusion hacks wouldn’t even happen in the first place.

That recent updates has parts two parts, one is the hot fixes Adobe or Lucee put out when a hack is discovered and they quickly put out something, and they both do a great job of that. They don’t cover up that there's been a hack they put out that hot fix and send out an alert.

The transparency sometimes feels like it can hurt the image of CF. Because every time Adobe gets the security exploit which isn’t very often to be honest, it’s very highly publicized. That’s good because you want people to know about it.

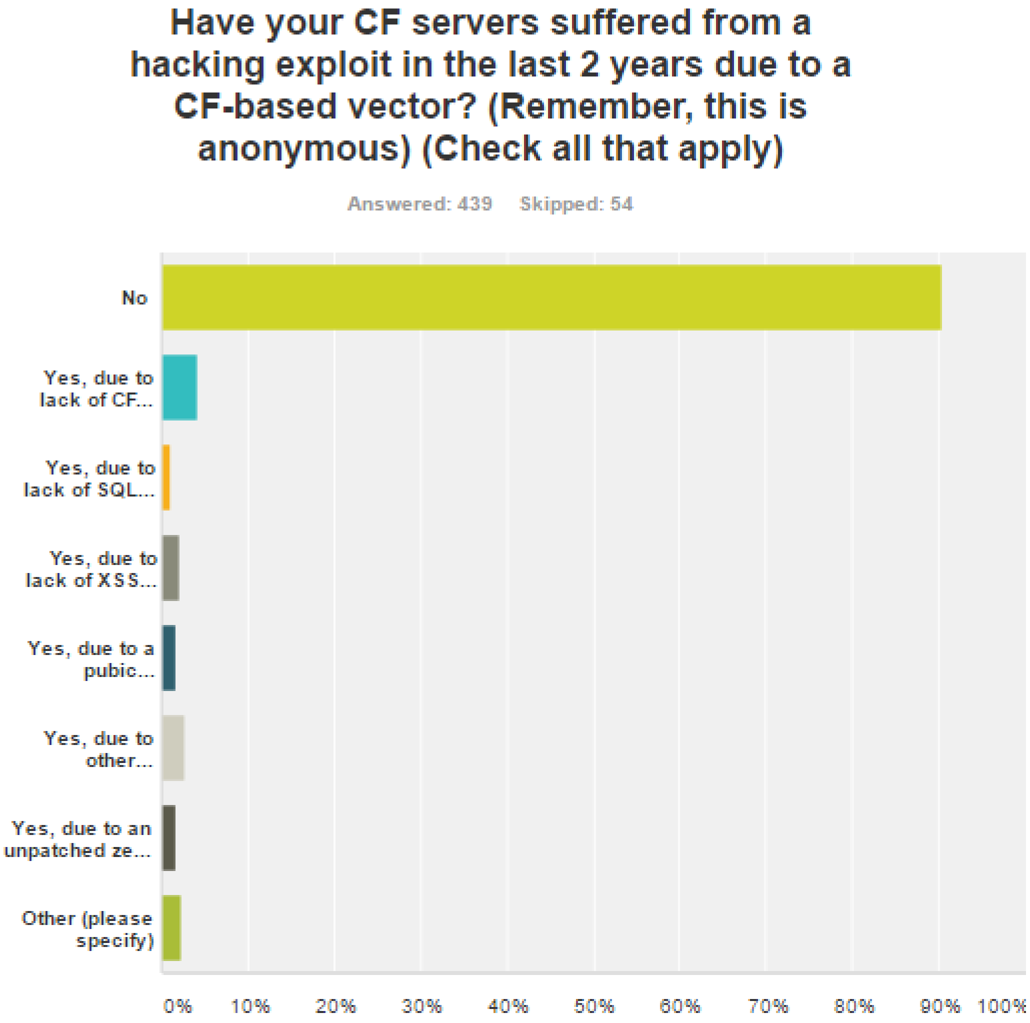
The secure profile feature is another good one; more people need to be using that.

Another security issue is that too many people are on older versions of ColdFusion that are insecure.

There are two parts to this. First of all you want to get the hot fixes as they come out and your registered CF user will get alert emails about those. But the other thing is you need to get off end of life ColdFusion which means CF9 or earlier and CF10 is going to go end of life in May 2017.

ColdFusion 11 and CF 2016 both have the secure profile. When you install ColdFusion as user it’s a production server, so many so many people forget just the obvious stuff like a custom air template for this site. They are putting all sort of juicy information for hackers and looking like newbs at the same time and secure profile sets a whole handful of settings just right off the bat that will make your server twice as secure. Also in ColdFusion 2016, say you are using one of the most commonest installations, they’ve now blocked access to the entire CFIDE Folder by default you can’t even get to it. The IIS connector won’t even hit it. The only way to get to it is to hit it directly through Tomcat but those things are way worth it. Because 99% of ColdFusion installs previously weren’t secure by default and most people didn’t lock them down after installing.

36. Have your CF servers suffered from a hacking exploit in the last 2 years due to a CF-based vector?



This was anonymous question so probably 90% of people haven't had a CF based hack but that means 10% of people have been hacked which is upsetting.

Overall it is good to see a lot of no's. A lot of people in the community who are not ColdFusion developers, they widely regard ColdFusion as just a very insecure platform, just any ColdFusion servers just immediately hackable and that are not the case statistically. ColdFusion has a very low number of exploits and this is a good graph. It's interesting to see out of the people who were hacked, what was the reason. The most answered one is lack of ColdFusion patching which isn't that surprising. That's one of the easiest ones to do. In ColdFusion now you can even configure it to literary email you when a new patch has come out. You'll get an email in your inbox that says here's an update and people who don't patch it, 4% of people who answered the question were hacked because they simply didn't install the updates.

Then other hacking methods of SQL injection, cross side scripting, holes where the admin is publicly available, other vulnerabilities in the code.

In the public administrator, it is also incredibly easy to lock down a lot of people just don't do it. The only people you can really feel sorry for is very small less than 2% that say they were actually hacked by actual on un-patched zero day in the engine itself. Adobe usually has patches available within two weeks or less when a zero day hole is found, so that's usually a very small window. The remaining people who get hacked just simply don't apply the updates at all.

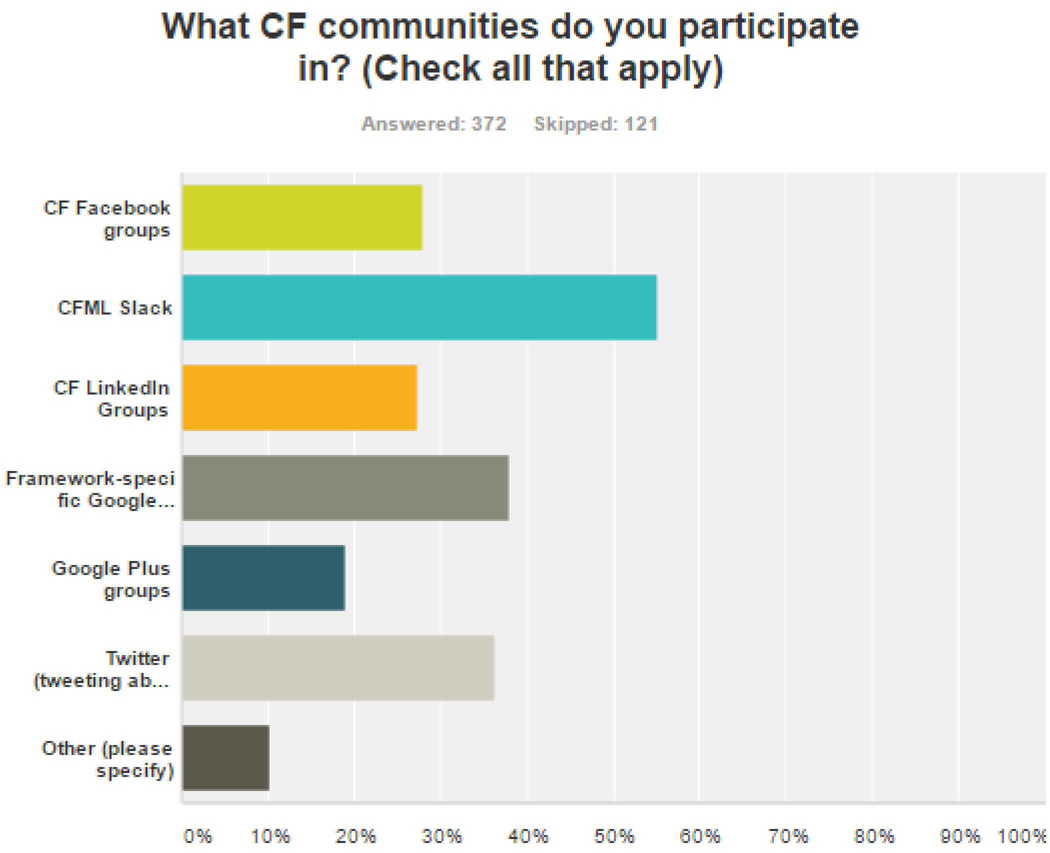
When you look at hacks that have happened against ColdFusion it's not day zero stuff, it's stuff that's been out for six months, a year, even longer, or they are using CF9 which doesn't receive hot fixes anymore because they went end of life and you're never going to get hot fixes for that. People just have holes in their setup.

Every company that has a segregated wall between developers and the dev ops guys who maintain the servers, they've got to have at least one person at the company whose responsibility is to keep up on updates. Whether it's Windows updates, Apache updates or ColdFusion updates, somebody has got to have a schedule where they go through and they check them all on a regular basis. It's the people who deploy a server and just leave it out there for years that they get hammered on this.

Usually, they were hacked about six months earlier and it's only when the hacker started uploading wares or pornography or whatever on their server they blew out the disc space and the server started crushing then they understood they've been hacked. This is actually very common. There's been some famous hacking exploits where it was only discovered months later. For example the South Carolina State government were hacked six months previously and it took them that long to discover that millions of confidential records had been stolen. People are sitting there thinking "I haven't been hacked" and that may not be true.

If you want to detect hacking earlier then use software such as Tripwire that detect malicious activity and notify you immediately.

37. What CF communities do you participate in?



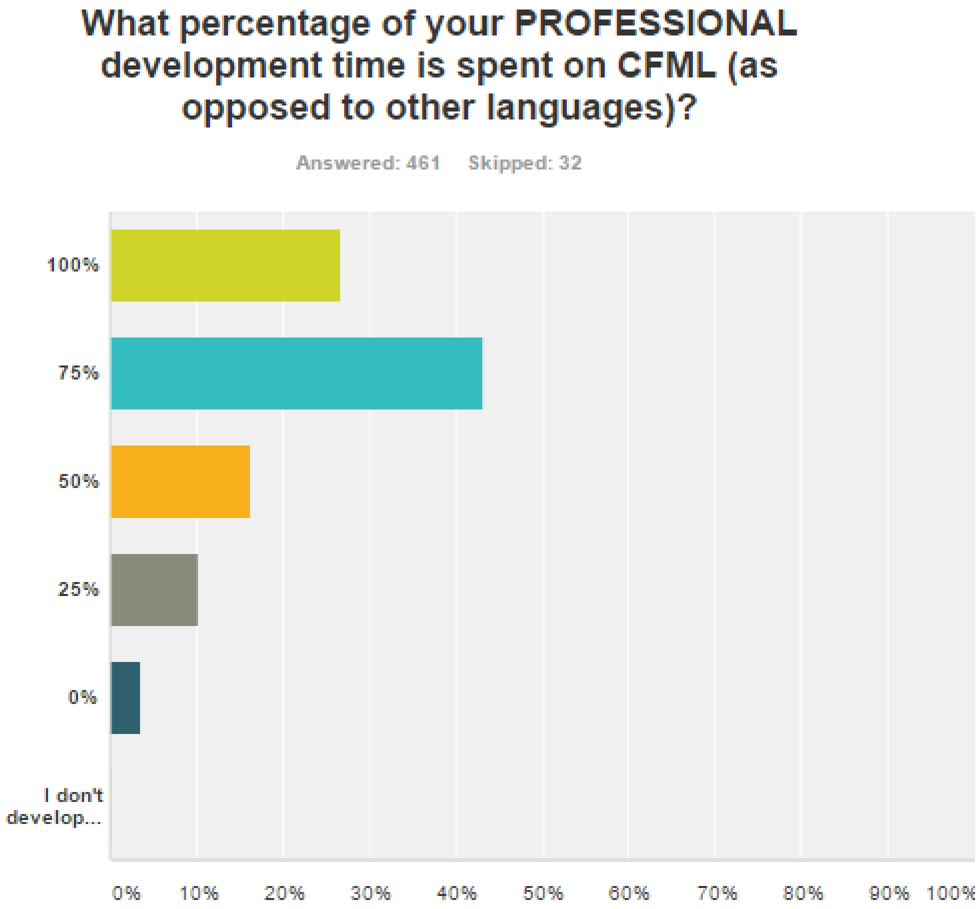
Most popular one is the Slack channel, which has a few thousand people in it and is very active.

That's a great place to share useful information or to get answers to questions. Then second here, there are some framework specific ones and then there are also some Facebook groups that are popular.

I think there are probably a lot of people who aren't in communities but those are the people who didn't take the survey because they weren't in the communities to know about the survey in the first place!

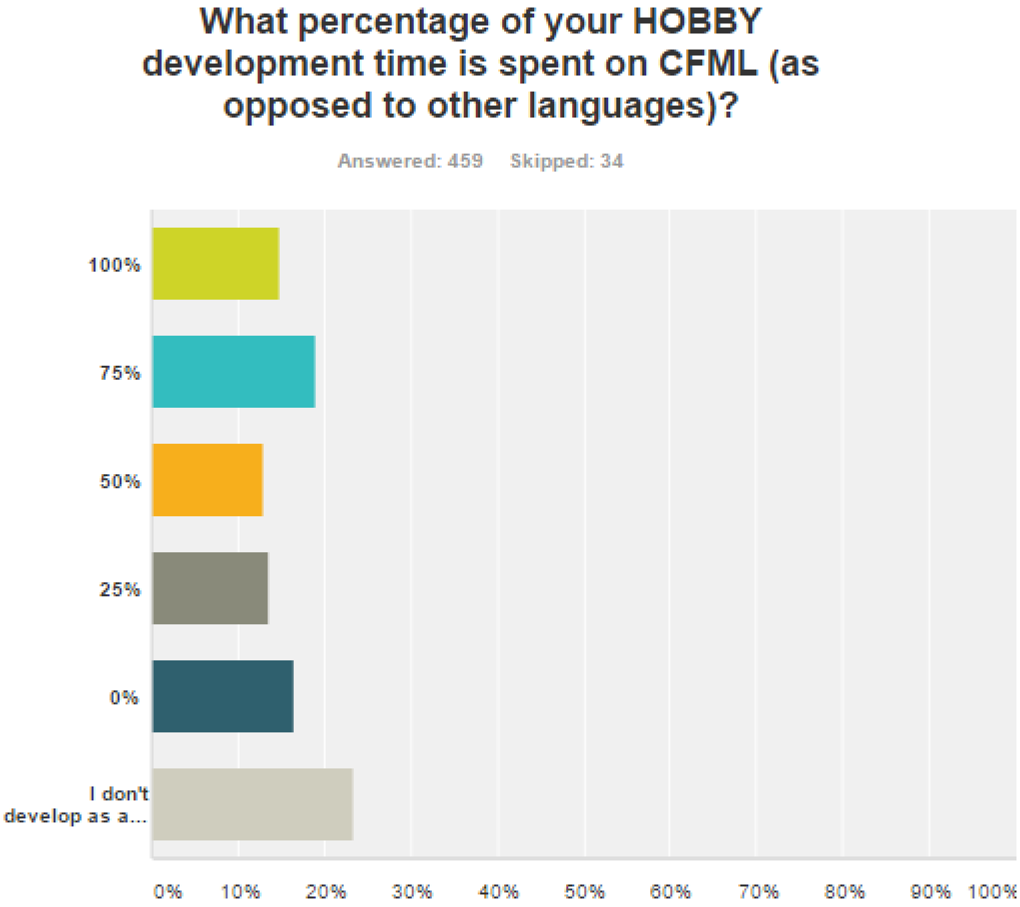
People either heard about it through one of these communities by other people sharing it there or they were on the TeraTech email list. There were quite a few tweets as well which we regarded as a community if you used that ColdFusion of CFML hashtag it's going to pick up traffic.

38. What percentage of your PROFESSIONAL development time is spend on CFML?



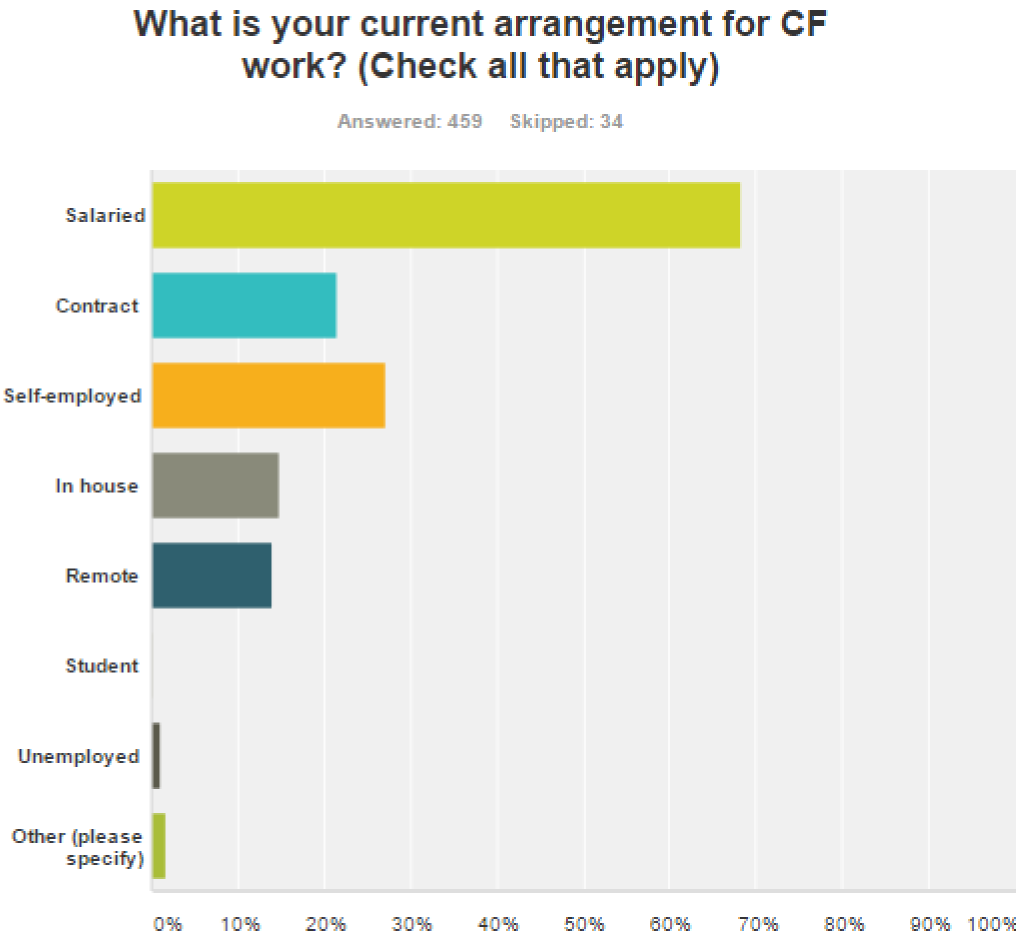
Most people taking this to focus on professional time and then hobby time (that's open source projects or your own thing).

39. What percentage of your HOBBY development time is spent on CFML?



It seems that most people doing CF professionally is mostly what they focus on. A lot of people doing CF in their hobby appear to focus on a lot of other languages in their hobby which is not a bad thing.

40. What is your current arrangement for CF work?



How do people work? Are they salaried, contract, self employed? Most people taking the survey were salaried but a fair chunk contract or have their own business.

A lot of people who are self employed probably also were contractors. They probably contract for themselves.

And, only 14% are remote.

There's a lot more people who would probably work remote if there were more virtual positions available. Both Ortus and TeraTech work remotely.

Then very few are unemployed which relates to that high demand for ColdFusion developers. Then hardly anyone said they were a student which I'd love to see that change because that's how we get fresh blood in the CF community.

41. Additional comments

One of the biggest ones are people looking for better support for ColdFusion and better marketing. There's always more that Lucee and Adobe can do in marketing. But Adobe and Lucee can never do the same grass roots marketing that you and I can do.

That’s what every open source language that competes with ColdFusion does.

There's a lot of languages that don't have a corporate entity to market for them. They were grown up out of just masses of people saying, "Hey I'm working on this really cool language called Blah”. Take a look at it and that's how they grew the community so that's something that everybody can do for sure to help that on their own.

We've got a number of people here who said they either love ColdFusion or bring CFML to the masses. A lot of enthusiasm about ColdFusion.

There were several mentions of Lucee CFML, people being excited about that.

Maybe we just need to tie those two things together. The people who are really enthusiastic about ColdFusion and the people who wish it was marketed better and get that message out. It's a wonderful language and someone can be proud of doing.

The ways people can do that can involve finding a local user group, even just an open source user group or a generic program and user group. Just being part of it and just talking about ColdFusion or going into a local conference that's not a ColdFusion conference and talking to people about what you do with CF. Or starting a local meetup group if there isn't one. They're very easy to start.
If you've got a town with enough ColdFusion developers, you can talk to Lisa Devork from Adobe and try to even start an official Adobe ColdFusion User Group or it doesn't have to be official, it can just be you inviting people to talk about the cool stuff that's going on in CF.

For people who don't have other people in town, the Southern Florida ColdFusion user group went virtual a few years back and anyone can attend their meetings online.

Charley Arehart also has the online CF meet-up and they don't meet in a regular basis but as needed when he has speakers, that's a group you can do online as well.

Southern ColdFusion One Resource has a round table format where people bring their coding problem they have so they don't have a speaker unless one turns up but they do meet every. That's run by JC Gomez.

So that's it for another ColdFusion State of the Union Survey 2017. Thanks for reading to the end!

About TeraTech

Expert ColdFusion Developers

CF_Revive is backed by the established ColdFusion consulting firm TeraTech. Founder Michael Smith is a long-time ColdFusion expert is well respected in the ColdFusion developer community.

We only work on custom ColdFusion websites and applications.

We've worked in all kinds of industries, including medium sized corporations, associations, education and government.

We're experts at taking over existing ColdFusion applications and giving them a new lease of life.

TeraTech has also built numerous complex web applications from scratch. If it's ColdFusion you need help with then you've come to the right place.

ColdFusion Alive podcast

ColdFusion Alive is the podcast hosted by Michael Smith, where he and his guest interviewees teach businesses how to stop wasting time and money hassling with their ColdFusion maintenance and start getting fabulous and reliable CF apps.

TeraTech Inc

The ColdFusion Experts

451 Hungerford Dr Ste 119

Rockville MD 20850

301.424.3903 Fax 301.762.8185

www.teratech.com

TeraTech