Ready Tester One? GO!

@melthetester

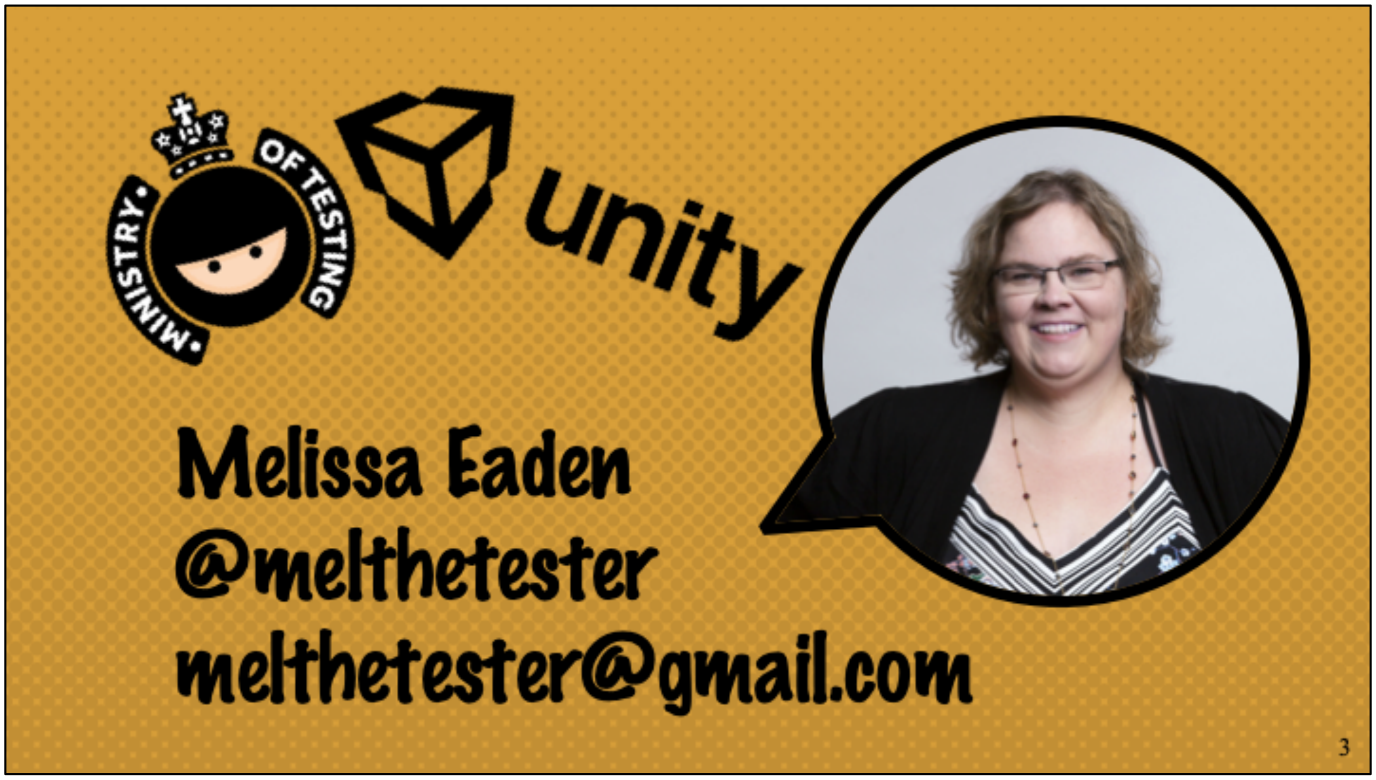
Good morning!

Hope everyone has coffee or a beverage of choice.

I make no apologies for borrowing from popular fiction for the name of this talk. It's something I actually do a lot of on my blog. I read something or see a movie and I think, how would a tester approach this problem. What problem does it actually help clarify for me, then I write about it.

I chose the title Ready Tester One, because I felt like it's very comparable to how I feel, and maybe how a lot of technologist feel when they approach their career, and this is especially true for those of us that are testers. We don't often get a defined career path. Companies and managers have a hard time defining that for a lot of us.

Hopefully, today, you'll get a lot of helpful examples and stories about how you can choose your career path and understand that you have more choices than you might have imagined.

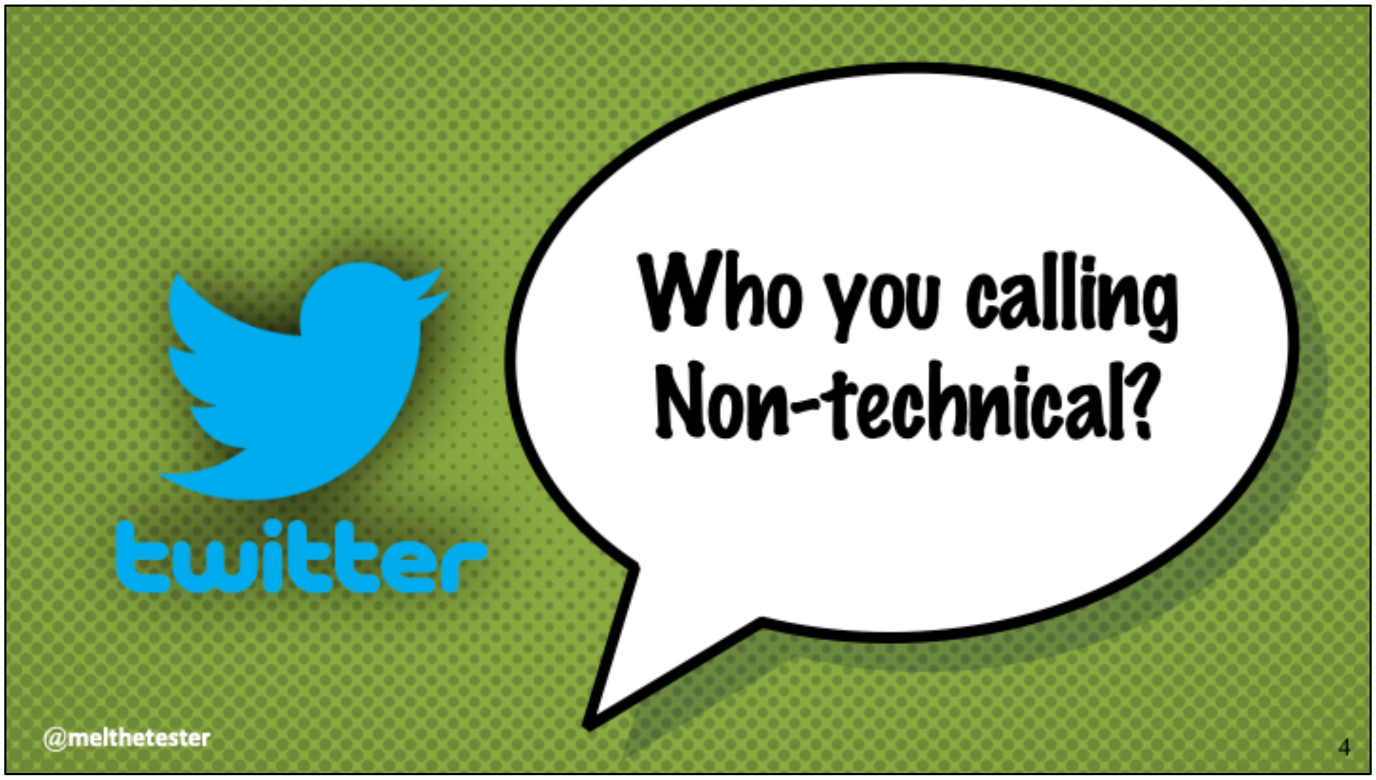


Melissa Eaden

@melthetester

melthetester@gmail.com

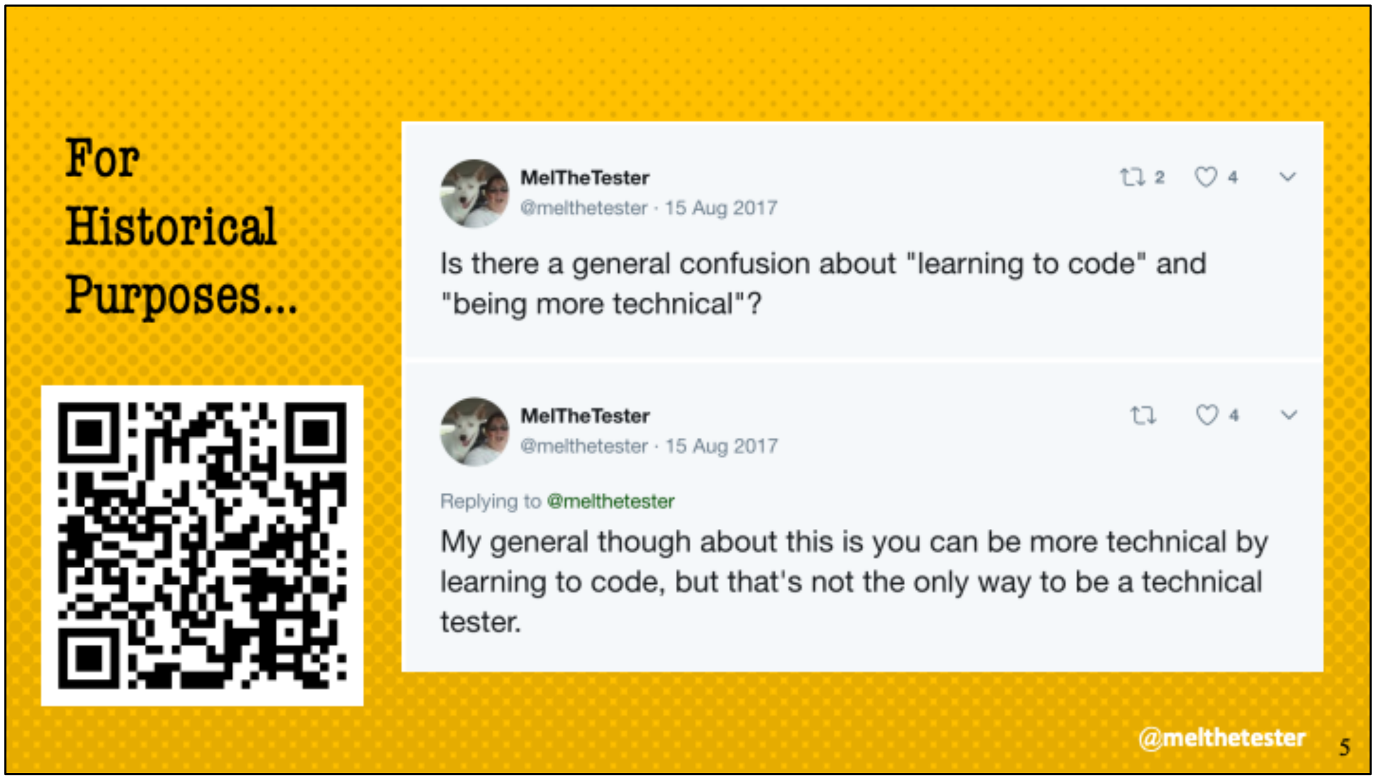
About Me!


twitter
Who you calling Non-technical?
@melthetester
4

Some ideas start in the weirdest of places. If you are not familiar with the testing community on twitter, I suggest following it, or joining twitter and seeing what testers are talking about. A theme that pops up often is the idea of technical vs non-technical. That testers are either one or the other. I have no idea where or how this got attached to testers, no one else in the industry really suffers from this odd notion. Program managers, business analysts, customer service folk all manage to be understood and classified correctly as folks that have technical skills either in business, writing, or computers in general. Testers however have managed to have this weird non-technical label attached.

The very idea that any tester is not considered technical is kind of a farce. There is a very real and different distinction between being non-technical and a non-coder. Again, somehow, everyone else has managed to be a non-coder, but technical, except testers. You have to code as a tester to be technical. Coding is a nice skill to have, but to say you have to be a coding expert, as a tester, to be considered technical is complete crap.

I have some theories as to why this happens to testers more often than other roles, but it’s a bit of guesswork at the moment.



For Historical Purposes...

MelTheTester @melthetester · 15 Aug 2017

Is there a general confusion about "learning to code" and "being more technical"?

MelTheTester @melthetester · 15 Aug 2017

Replying to @melthetester

My general though about this is you can be more technical by learning to code, but that's not the only way to be a technical tester.

@melthetester 5

The tweets….
https://twitter.com/i/moments/947154288698150912

Organized testing goes as far back the introduction of the scientific method. Lots of testing happens all the time, by just about everything on the planet.

Software testing seems to have started around the same time as the Mark I and Alan Turing with his Turing Test. It makes a lot of sense that testing came shortly after the first hardware/software combo comes into existence.

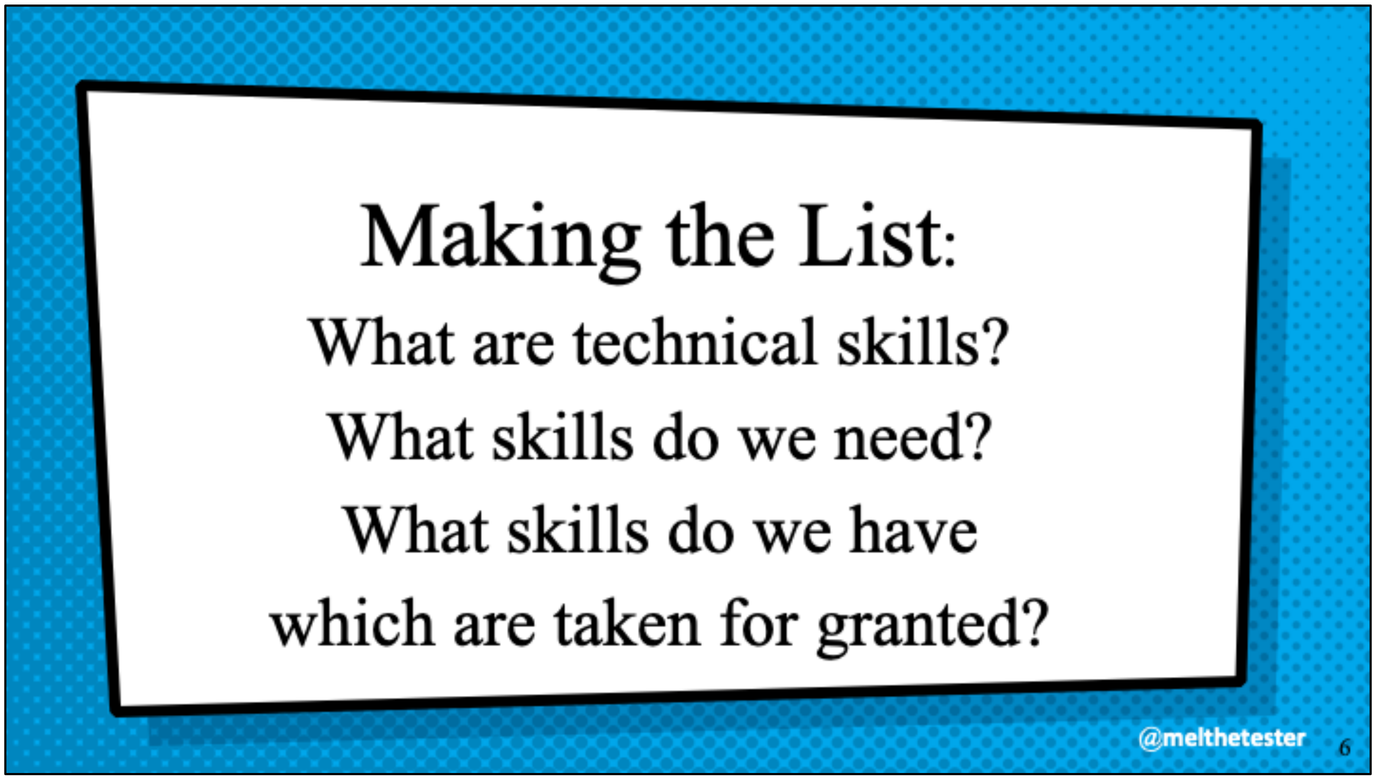
The next big leap into testing is 1957 where Charles Baker describes testing as a separate activity from debugging. Then, Gerald Weinberg forms the first ever testing team in 1958 for Project Mercury.

(link: http://www.testingreferences.com/testinghistory.php)

It's a tricky topic, testing, and being a QA or a tester are tricky titles. What does that actually mean? I call myself a tester, but I'm also a technologist. I work with technology. I have written code. I have a whole folder of the stuff on my home computer. I dabble, I understand how something works, code-wise, and then I tinker or play with it.

That skill doesn’t make me technical. What makes me technical is how I put that skill together with all the other skills I have.

This whole thread caused me to think about how I learned my skills, what I learned, and what I still need to learn. It’s a topic that keeps cropping in for me, so I wrote about it.



Making the List:

What are technical skills?

What skills do we need?

What skills do we have which are taken for granted?

@melthetester

6

Originally I had a working list of skills by levels I had created for a client because they were having a hard time finding talent with the right skill set.

I wanted to explain how I evaluate myself and what I thought we needed as skill sets. I started with my career chronologically, made a list, rearranged them again, made another list.

The original version had three levels. It eventually expanded it to five after I thought about it some more after the twitter thread I showed you. Once I knew what I was capable of doing, I looked for others I worked with and pulled skills from them as well.

Specializations are a thing, and they are much broader and deeper for testers that people realize.

-Automation is one… but if it's just UI, then that's only one space. What about services automation, pipeline automation, tool testing development and maintenance, automation design/architect.

-Performance & Load testing have a whole lot of things you can focus on

-Security testing is a whole sphere of techniques and tools

-Accessibility testing

-Mobile testing especially if you are dealing with more than platform, services, devices, then accessibility, on top of any native pay features, and then security…. On top of automation.

It’s an endless buffet of things you need to learn. The more I thought about it, the more the list grew and changed.

LEVELING UP!

@melthetester

https://testingandmoviesandstuff.blogspot.com/2017/09/ready-tester-one-go.html

Let us do a little flashback though - I got a lot of my ideas from gaming, and games in general.

I love playing games. I’ve played a great number of tabletop games, board games, video games… one of my favorites right now is Don’t Starve.

But once upon a time, I used to be a larper. LARP is Live Action Role Play.

LARP!?

Vampires, Werewolves, and Fairies, Oh MY!

@melthetester

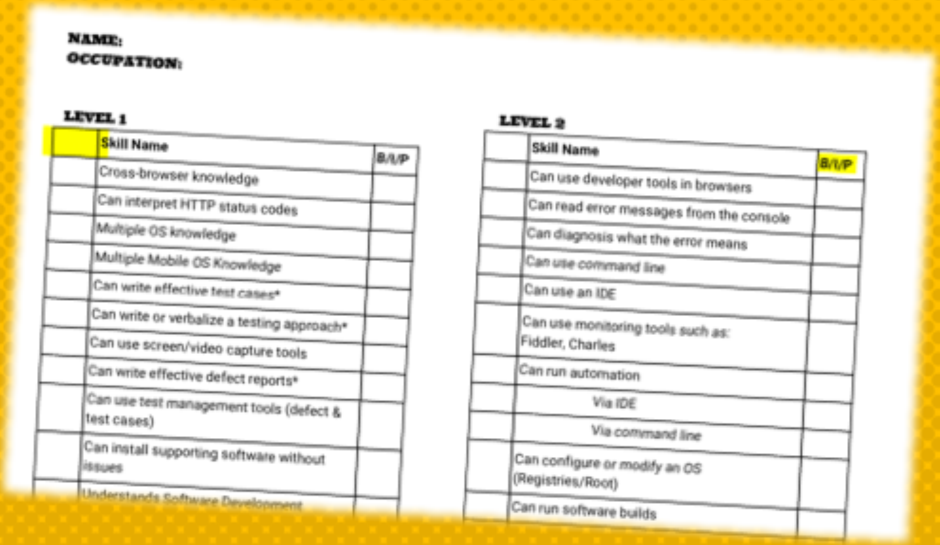
I spent about eight years playing LARP (or Live Action Role Play) games created by White Wolf. The general idea is that you would make a character, based on some rules and roles, then act the character out. I did that for a number of years, and then I moved to being a storyteller for LARP games. Storytellers are the folks who take all these wonderful, crazy characters people invented and then direct them through plot lines and stories either created by the storytellers as a main plot, or created by characters interacting with one another.

How does this equate to a career? Well, if you think about your career like a character sheet, the leap isn't that far. You have some basic talents and skills. You learn some specializations and keep building on your "character" as time goes on. As a storyteller, through the stories I tried to tell, I directly or indirectly shaped how people added talents and skills to their characters. Interactions with other characters did this as well. This is akin to what a manager or business might do for an employee through mentoring and/or career development, or what might happen if you find a mentor to teach you certain skills.

The problem is, for testers, businesses and managers have a hard time figuring us out and what exactly we need to do and why we need to do it. There are no clear career paths for testers, in testing with an idea that you grow your skill sets - it's out there for sure, but companies are somewhat poor at identifying them. It comes from the idea that "any tester will do" - because testers are generally not specializing in a code language, or our talents are not tied to a particular tool set, we are seen as interchangeable units. Sometimes.

Automation Testing is where this divide gets interesting. While Generalist Testers (testers that can do manual, automation, service, usability, design, security, data… etc) sometimes take longer to be seen as value adds, testers that squarely put themselves in an automation or SDET role are seen as immediately valuable by companies because they are seen as producing SOMETHING. They have tangible code with visible displays of speedy interactions with UIs. It’s the flash and the grandeur of automation that people are dazzled by, and because of that, they sometimes devalue other skill sets testers have. I’ll make a note here that I’m not anti-automation at all, but I would find my job pretty boring if all I was doing was automation all day long every day.

Career Character Sheet!



NAME:
OCCUPATION:

LEVEL 1

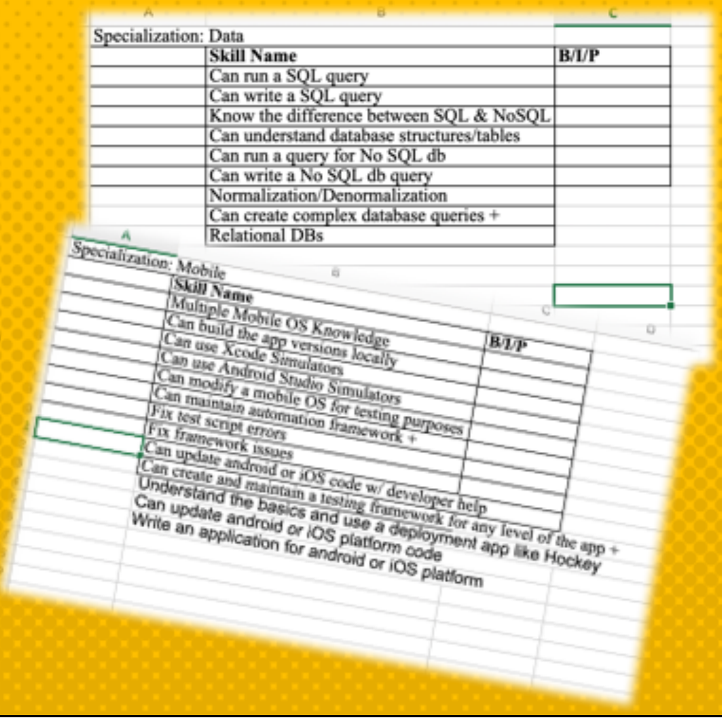
	Skill Name	B/I/P
	Cross-browser knowledge	
	Can interpret HTTP status codes	
	Multiple OS knowledge	
	Multiple Mobile OS Knowledge	
	Can write effective test cases*	
	Can write or verbalize a testing approach*	
	Can use screen/video capture tools	
	Can write effective defect reports*	
	Can use test management tools (defect & test cases)	
	Can install supporting software without issues	
	Understands Software Development	

LEVEL 2

	Skill Name	B/I/P
	Can use developer tools in browsers	
	Can read error messages from the console	
	Can diagnosis what the error means	
	Can use command line	
	Can use an IDE	
	Can use monitoring tools such as: Fiddler, Charles	
	Can run automation	
	Via IDE	
	Via command line	
	Can configure or modify an OS (Registries/Root)	
	Can run software builds	

This is version one of the character sheet I came up with. It’s on my github, so don’t worry if you want a copy of it. I liked it. It focused on core skills and ideas based on levels. This first version has 5 levels. The more I thought about it, the more I wanted to expand on it…

Advanced Career Character Sheet!



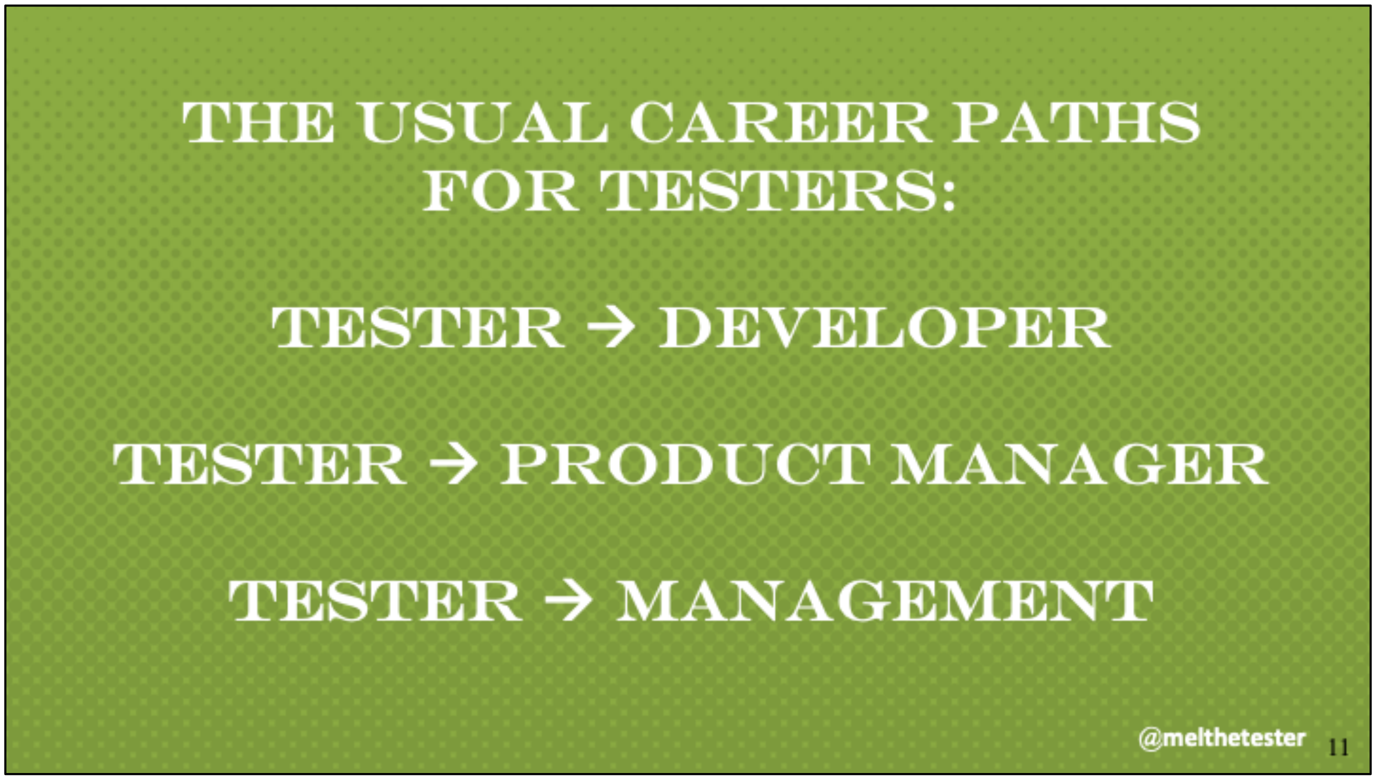
Specialization: Data

	Skill Name	B/I/P
	Can run a SQL query	
	Can write a SQL query	
	Know the difference between SQL & NoSQL	
	Can understand database structures/tables	
	Can run a query for No SQL db	
	Can write a No SQL db query	
	Normalization/Denormalization	
	Can create complex database queries +	
	Relational DBs	

Specialization: Mobile

	Skill Name	B/I/P
	Multiple Mobile OS Knowledge	
	Can build the app versions locally	
	Can use Xcode Simulators	
	Can use Android Studio Simulators	
	Can modify a mobile OS for testing purposes	
	Can maintain automation framework +	
	Fix test script errors	
	Fix framework issues	
	Can update android or iOS code w/ developer help	
	Can create and maintain a testing framework for any level of the app +	
	Understand the basics and use a deployment app like Hockey	
	Can update android or iOS platform code	
	Write an application for android or iOS platform	

The advanced version! It's divided into core skills, and the specializations, with skills associated with specializations. This is where it does get a bit nerdy in the fact that I'm like OK – I'm declaring I'm a tester – like I would declare that I'm playing a Halfling… which is pretty much what I play for D&D, and then all the specializations are like classes. Instead of wizards and druids, you have data analysts and performance testing. Best of all – you can multiclass! So maybe you are an automation specialist with a couple of levels in management or leadership… or you've been working with mobile apps for a while now, and now you're branching into ops because you need more robust mobile testing…



THE USUAL CAREER PATHS FOR TESTERS:

TESTER → DEVELOPER

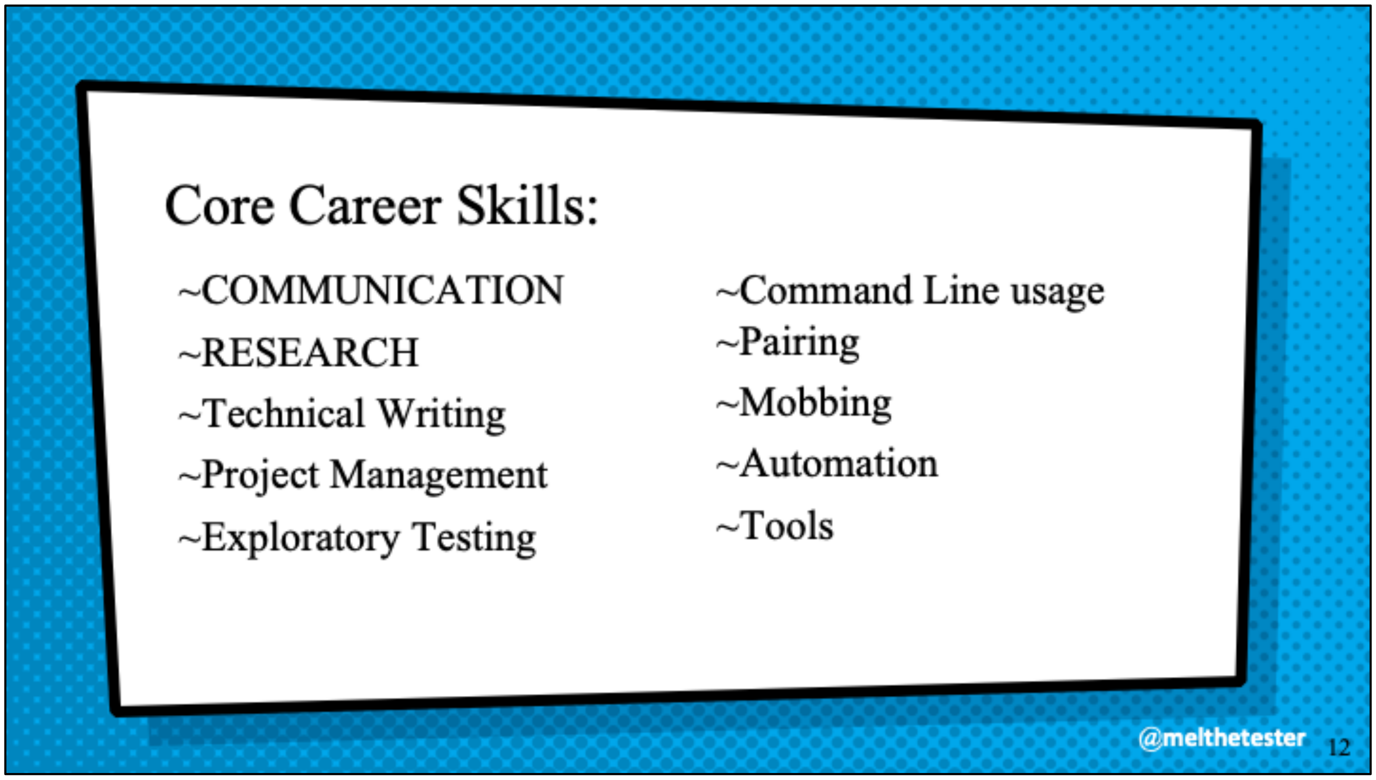
TESTER → PRODUCT MANAGER

TESTER → MANAGEMENT

@melthetester 11

Typical career paths for testers --- tester to developer, tester to product person, tester to management… unfortunately a lot of places use testers as junior roles into other areas.
What if there is another, more complex and helpful way to develop testers and these are by gaining specializations.

However, before you can get into a particular specialization, we should take a look at core skill sets, basic skills that every career character sheet should have that allow someone to build into specializations.


Core Career Skills:

~COMMUNICATION
~RESEARCH
~Technical Writing
~Project Management
~Exploratory Testing

~Command Line usage
~Pairing
~Mobbing
~Automation
~Tools

@melthetester

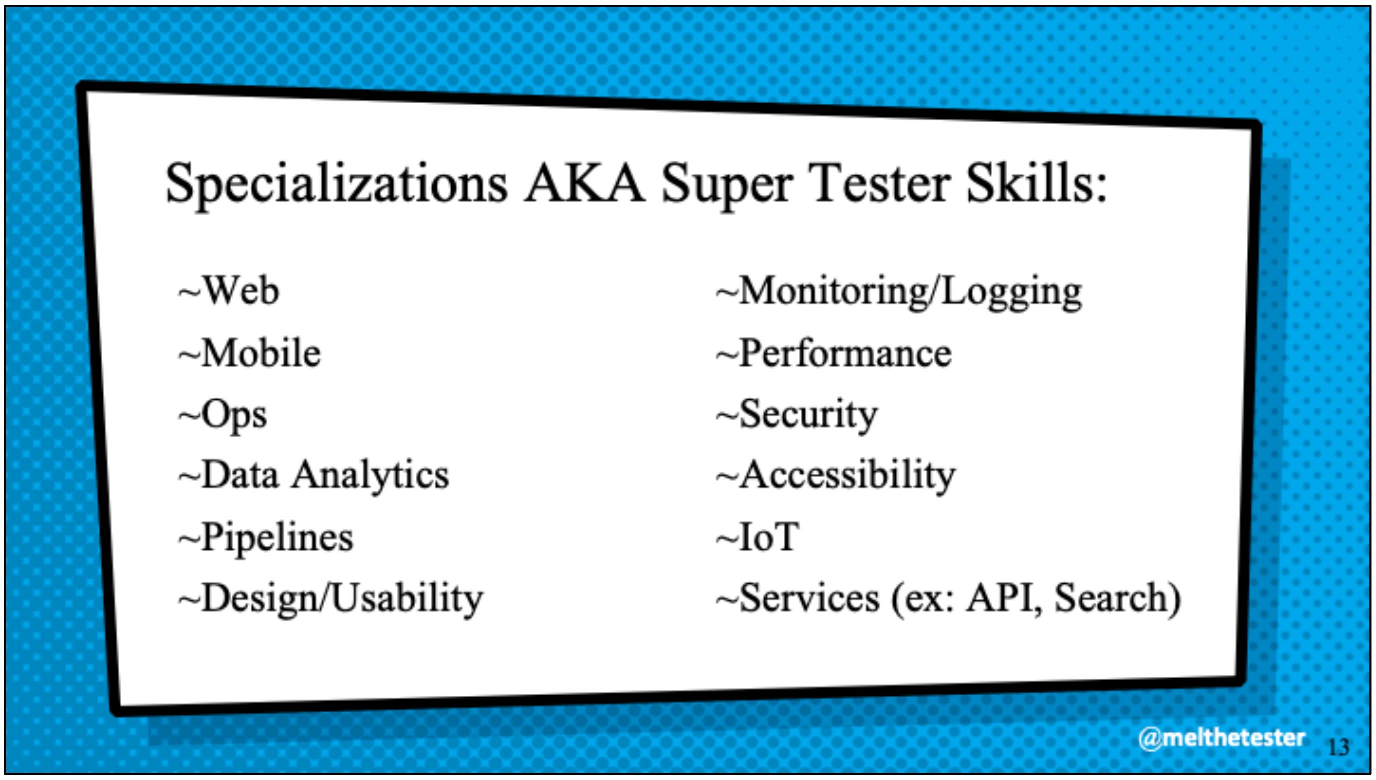
Communication is in caps because it's probably the most important skill any tester has and it's what makes us good at what we do. It's the hardest skill to master, and it's the easiest to screw up. Everything we do is literally based on this one skill.

Research is in caps because I feel like this is the second skill you have to have for testing. Research addresses things like: gathering test scenarios, user data, creating a targeted testing matrix based on analytics. Domain knowledge.

There are companies that are creating whole new job titles just based on domain knowledge. This could be its own specialty, but I think you to be a good tester, you have to understand the domain you are working in and why people are using the software you are working with, the driving factors, the motivations, and then you need to understand the competitors. There are even more specialized areas of domain knowledge for search functions called relevance engineers.

I have rarely ever worked in the same domain twice. I always start any assignment or job with a ton of research.

Some might be surprised that I've put project management as a core skill, but it is. You have to manage stories, you need to manage defects (whether that's talking with your team immediately or reminding them of the backlog), you should be managing your time around the project or projects you are involved in. This skill can turn into its own job too and I've learned a lot from other product managers, project managers, and business analysts which I’ve used to make my job easier, to understand things better, to manage expectations and ideas more.



Specializations AKA Super Tester Skills:

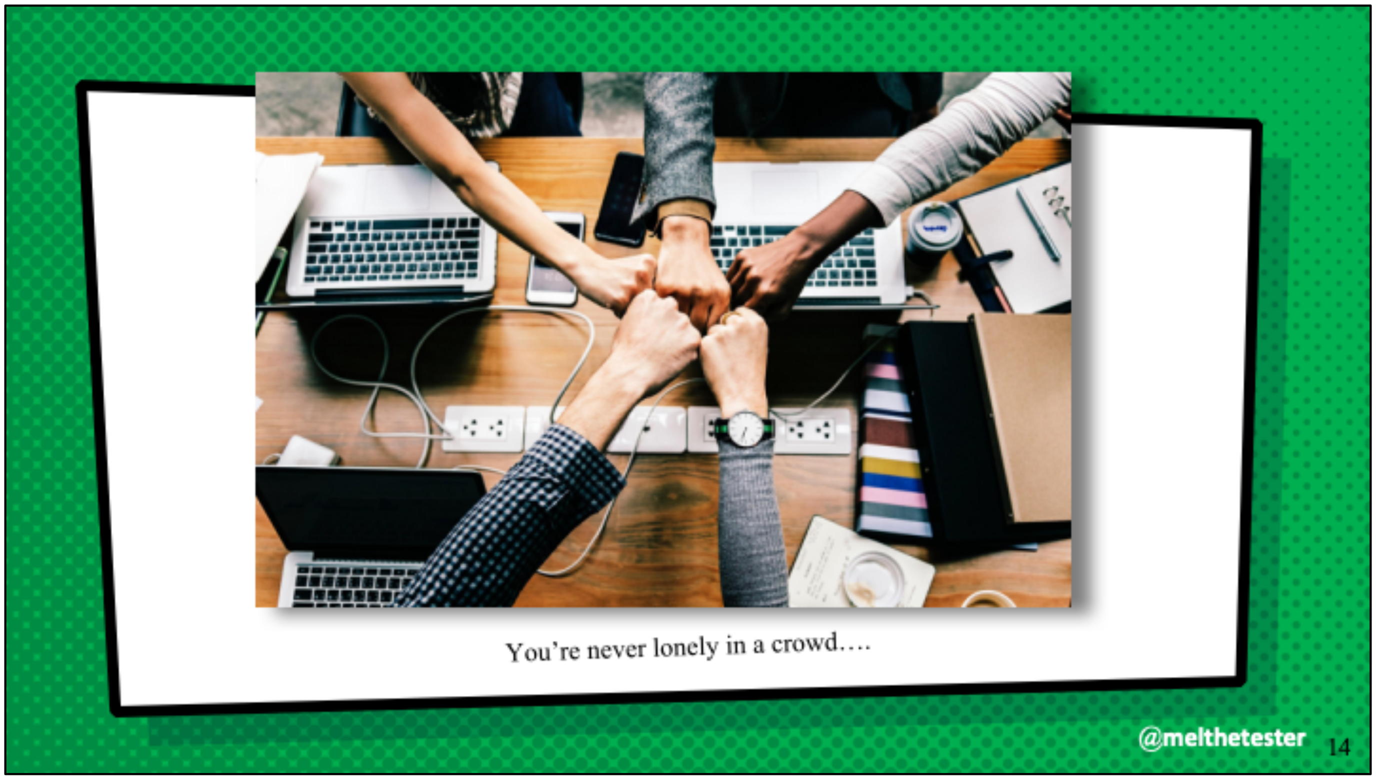
~Web	~Monitoring/Logging
~Mobile	~Performance
~Ops	~Security
~Data Analytics	~Accessibility
~Pipelines	~IoT
~Design/Usability	~Services (ex: API, Search)

@melthetester

These are the specializations I've identified, and it's not even a complete list. These don't even really address what happens when you get into combinations of things like Mobile Data, or Mobile Pipelines. Web design, or IoT logging… There is a complete ecosystem out there with developers writing and building things. There are testers out there which understand these combinations and environments and put these specializations together and keep leveling up skill sets to become experts in testing all combinations of things and automate them as well!

What we need to do as testers, as managers of testers, as hiring managers: understand what you are looking for in regards to testing. Understand the combination of skills that you think you might need at the basic level, then understand what specializations might be necessary to do the job. It's impossible at times to find the perfect person, however, if someone has the willingness to learn, to continue learning, to teach others, I promise you they will learn everything you throw at them, challenge themselves to do more and work harder, and then when they leave, they will look for another challenge, and you might be crying because you won't have realized what you had until that person is onto their next job.

I say all this to say, don't pigeon hole your testers. The number one trait of most testers - curiosity. Present everything as a mystery or a challenge to be solved, and with competitive pay and a good cultural working environment, they can do amazing things!



You're never lonely in a crowd….

@melthetester 14

You might be wondering… how do I get all the skills, how do I find them, how do I… etc…

Well, crowdsource them! You have networks and contacts you probably have that are more than willing to help you along your journey to learning any of the previous skills mentioned. I wrote an article about it, and how you can get a network going, where to look for them… if you don't have a twitter account, get one. The testing community and dev communities are very active. Get out to local meetups. Get names from people. Read books then contact the author; ask questions. Get a book club going, a study group, get linked up on LinkedIn. Read articles people have written then contact the author of the article, learned who they learned from. Go to conferences like this one and collect business cards and stay in touch with people. Start writing a blog, start speaking, get visible… ask for help.

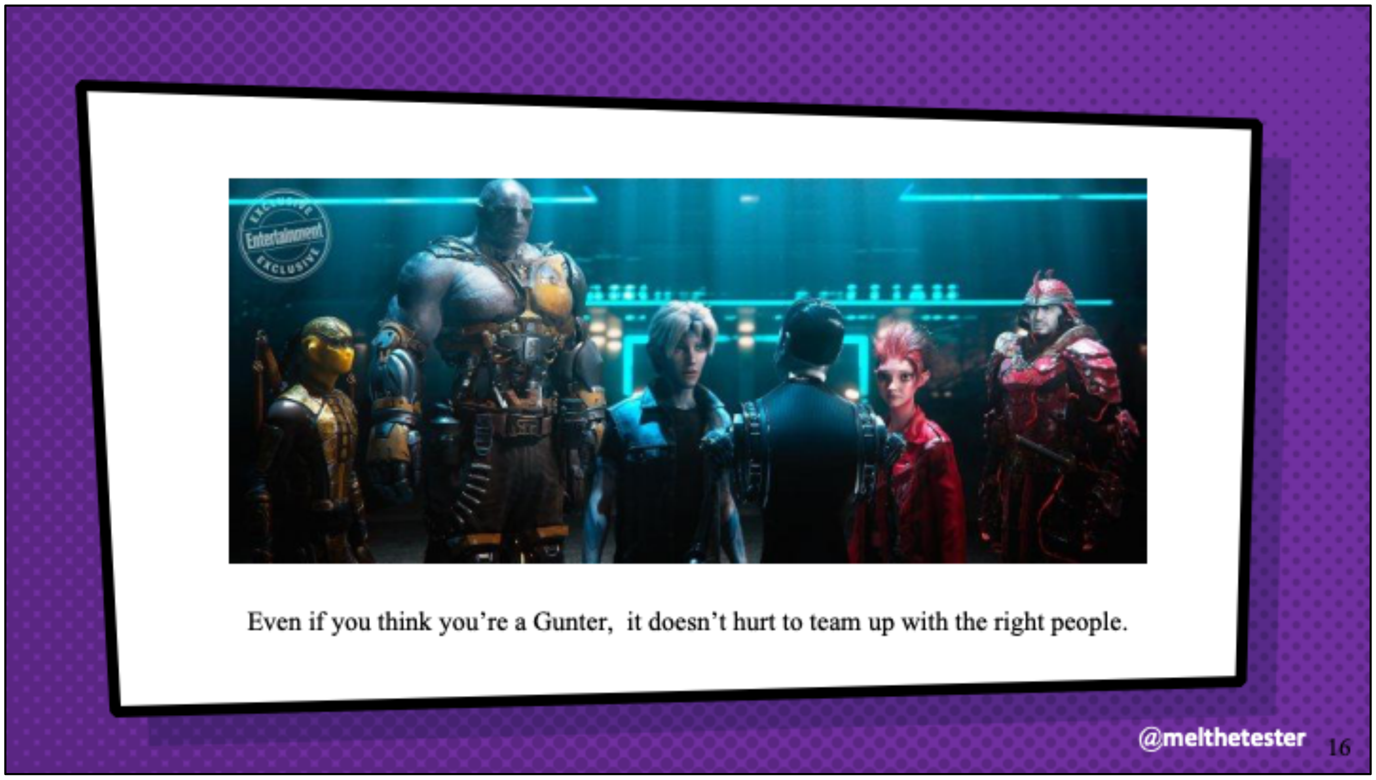
I've told this story a couple of different places. I was working on some postman automation, and I was running into a bit of a problem with an if/else statement. This was frustrating the crap out of me and I couldn't get it work. I was trying to get the test to skip if it ran out of variables to use. If it realized it didn't have any variables left, it would mark the test as skipped and give the response: No Time slots left - Test Skipped. This gave us a heads up in the CI pipeline and the test environment that failures were not because of an issue, but because we were testing too much. Never a bad thing really, but it's a pain if you for pipeline fails for something as simple as a missing variable which is supposed to be missing.

I reached out to Danny Dainton to ask him about the problem. I knew him from my network. He’s super good with postman and has a great tutorial up on github. I looked through his tutorial and found something useful, but it wasn’t what I needed.

By the way, here is a link to his tutorial.

I pinged him and asked a few more questions. He was awake to my amazement! I was on the East Coast and he was in the UK. I explained the problem without giving any state secrets away and he worked through it with me. We passed code snippets back and forth over Slack. He figured out how to handle my variable problem and I taught him that if/else still worked in Postman. It was fast, collaborative, and really rewarding for both of us.

I wouldn't have ever been able to do that without a network around me. Danny was cool enough to work with me in real time, but even if it hadn't of been real time, I'm sure he would have worked with the problem, giving me a few tips and advice to get me going, which would have worked out too.



Even if you think you're a Gunter, it doesn't hurt to team up with the right people.

@melthetester

The tech community is changing. We might all be a little introverted, but we are willing to help each other, without the snark and the attitude of previous decades. Shit is way too complicated for one person to do on their own all the time. You might solve parts on your own, but when you get stuck, getting perspective and help is so valuable. Stack Overflow is OK, but I suggest getting to know someone who you can ask.

I recently gave a workshop with Lisa Crispin in Boston. It was the first time I had done a workshop and the whole process was new to me. Lisa was great walking me through a lot of the steps we needed to do to get to a viable workshop at the conference and then presenting it with me. Others I know who have done workshops before gave us great feedback, and because of that, I'm sure we'll do the workshop again. I'm also sure we have a good second version in the works.

You can crowdsource anything, really.

Let's do it right now!

Talk To Your Neighbor

@melthetester

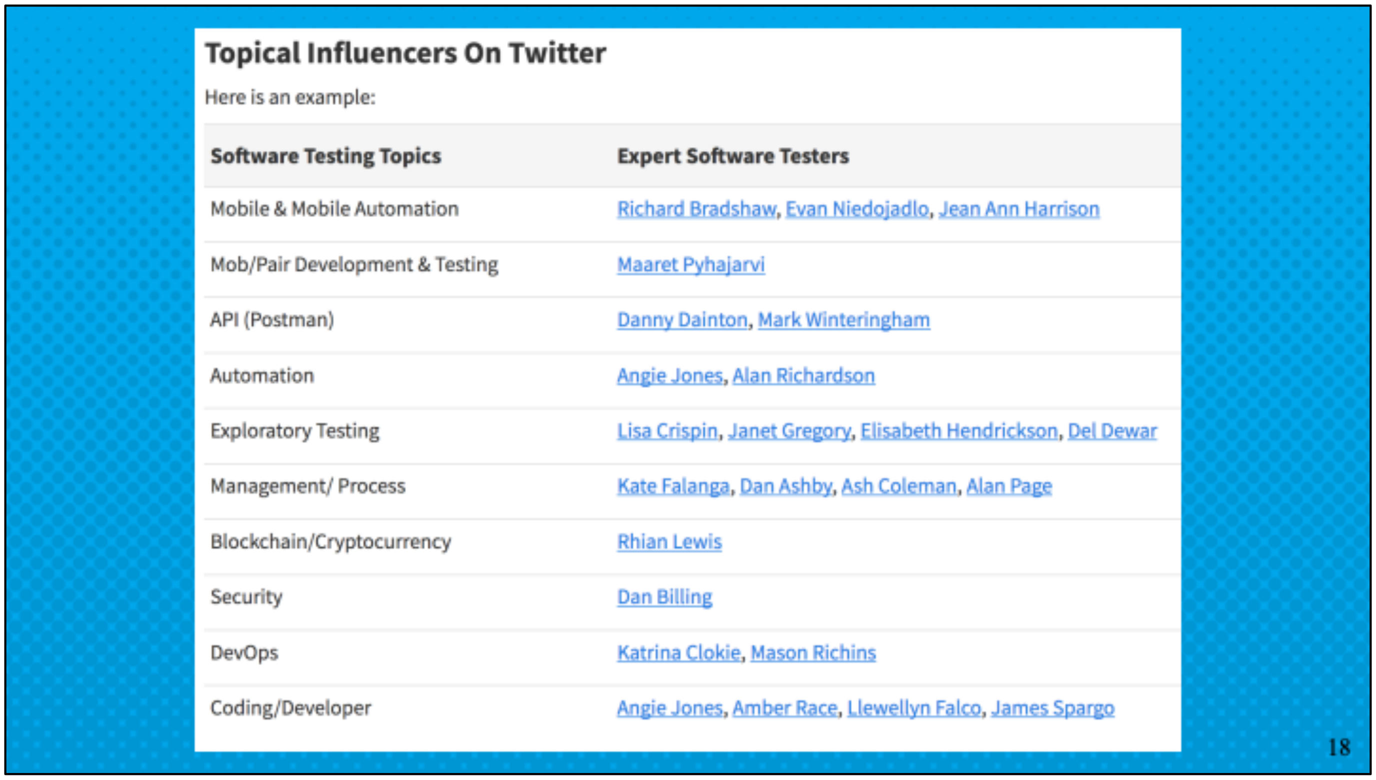
If you don't want to do this right now, it's ok introverts, just say no thank you…

Everyone else, get a name, get a topic or project, get a business card or a twitter handle.

You have your first name of a network!

What I'd like you to do next is think about your domain – what area of knowledge are you working in right now? Add that to your nametag. Every time you go to an event, whether it's a meetup, or a conference, or a workshop, add something about your current domain. This makes for a great ice breaker, and you could end up finding someone working in the same domain you are and you can swap information!

So now that you have that, would you like a few more people in your network?



Topical Influencers On Twitter

Here is an example:

Software Testing Topics	Expert Software Testers
Mobile & Mobile Automation	Richard Bradshaw, Evan Niedojadlo, Jean Ann Harrison
Mob/Pair Development & Testing	Maaret Pyhajarvi
API (Postman)	Danny Dainton, Mark Winteringham
Automation	Angie Jones, Alan Richardson
Exploratory Testing	Lisa Crispin, Janet Gregory, Elisabeth Hendrickson, Del Dewar
Management/ Process	Kate Falanga, Dan Ashby, Ash Coleman, Alan Page
Blockchain/Cryptocurrency	Rhian Lewis
Security	Dan Billing
DevOps	Katrina Clokie, Mason Richins
Coding/Developer	Angie Jones, Amber Race, Llewellyn Falco, James Spargo

Here is an example of my network. This took some time getting to know different people online, having others refer me to folks, or working with folks via the Ministry of Testing. I have been seeking these folks out and saving up names and contacts for when I might be able to ask questions. I'm also available for any of these people too. I might not have anything the need right this minute, but you never know. You start talking about what you've accomplished and people who have helped you, sometimes that comes back in exciting ways.

(For API - JoEllen Carter (@testacious))

...which is always changing but this list from an article I wrote for Ministry of Testing about Crowdsourcing your Learning…

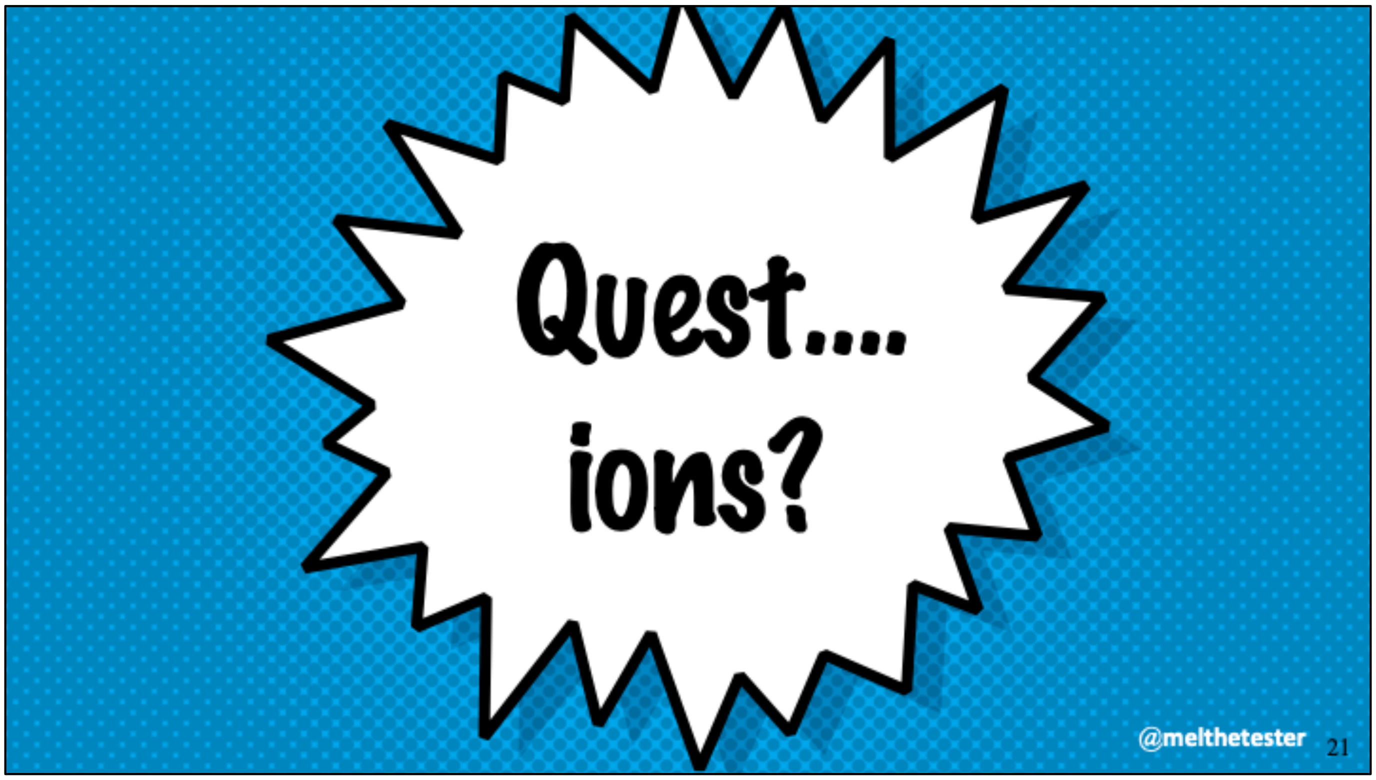
Here is the mega QR code for that article.
https://ministryoftesting.com/dojo/lessons/crowdsourcing-your-learning?s_id=12181

The idea for this article can be linked back to several times I've asked for help or sent out a question into the universe and someone answered. Or what usually happens is a lot of people answer. We have all can have these awesome social networks. We have a wealth of knowledge sitting online. I might sound like captain obvious right now, but when I've talked about how I've created a network through different social medias and contacts people seem surprised. If people are willing to help with so many other aspects in life, it makes sense to me that a lot of people are willing to help you figure things out in a technical aspect as well.

Another great example of this is a guy named Alan Richardson. He’s also in the UK. He has a bunch of wonderful online books I would recommend to anyone. He also recently started a Patreon. If you aren’t familiar with this platform, you should check it out. It’s like having a monthly subscription to awesome stuff and it’s easy to sign up and you can even moderate your payments based on how much you want to support someone. Alan is actively creating content to teach people about all kinds of topics related to coding and testing.

I’ve had wonderful interactions with him online. I did a review of one of his books and he replied!

Creating networks and relationships both internal and external to our jobs help build up the i/o of information. It’s like any game, you have to first understand the rules, then the unspoken rules, then look for information to solve the problem you are presented with, whatever that problem might be. It’s an adventure! It’s like opening one of those Choose Your Own Adventure books but instead of the result being death or something equally lame, you get knowledge, and it’s infinite. Or you can start over and learn a new path. That’s the awesome thing about technology, there is always something new to learn and someone to learn it from. You only have to find them.



The Quest awaits, are you ready tester one?

Choose Your OWN ADVENTURE!

@melthetester

Would you like to fill out your career character sheet? Check out this link and feel free to give me feedback and ideas or a submit a pull request even.

<TestingUy>

¿PREGUNTAS?

¡MUCHAS GRACIAS!

Melissa Eaden

melthetester@gmail.com

@melthetester

unity

13 y 14 de mayo, 2019

testinguy.org | @testinguy | #testinguy