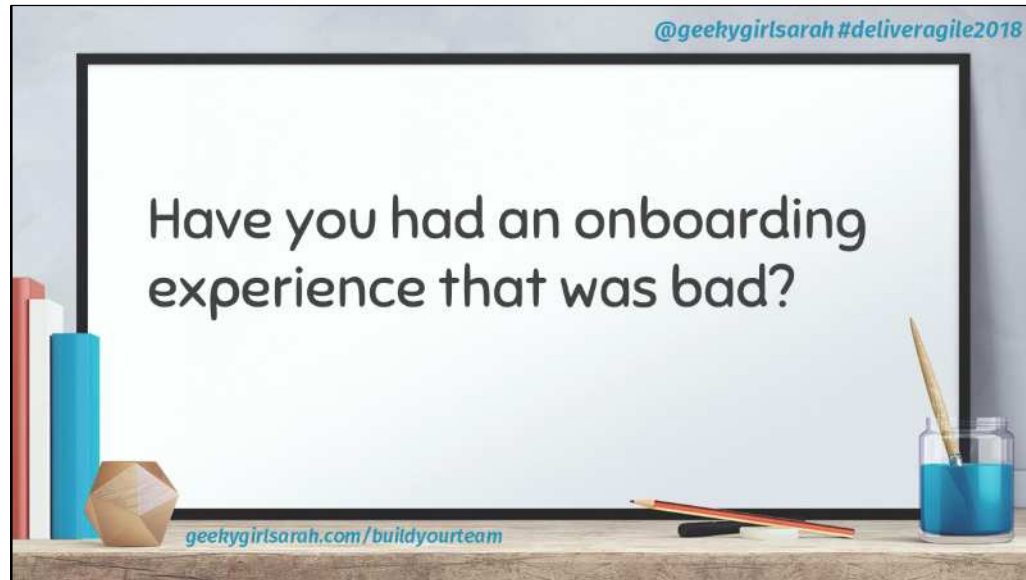
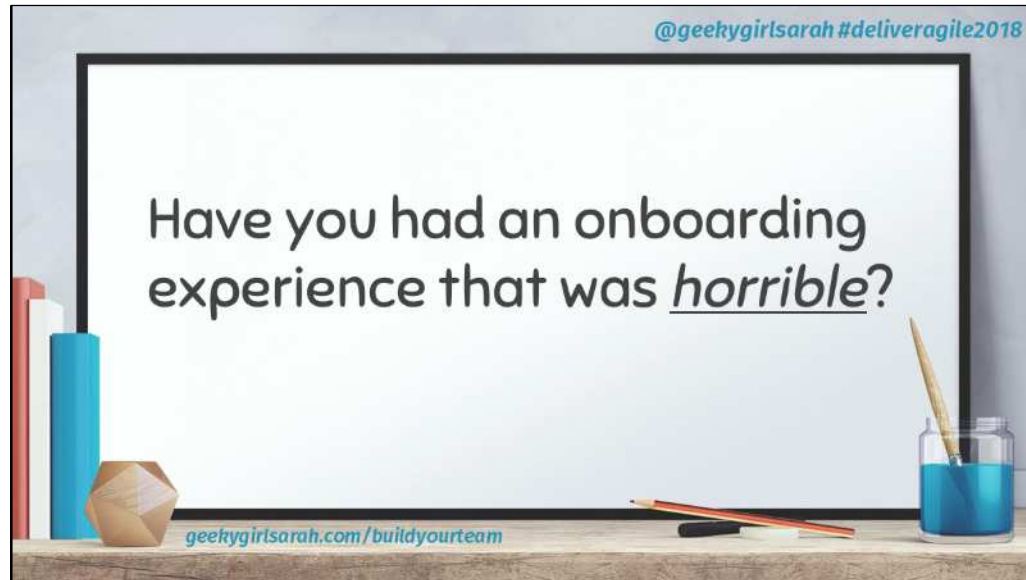




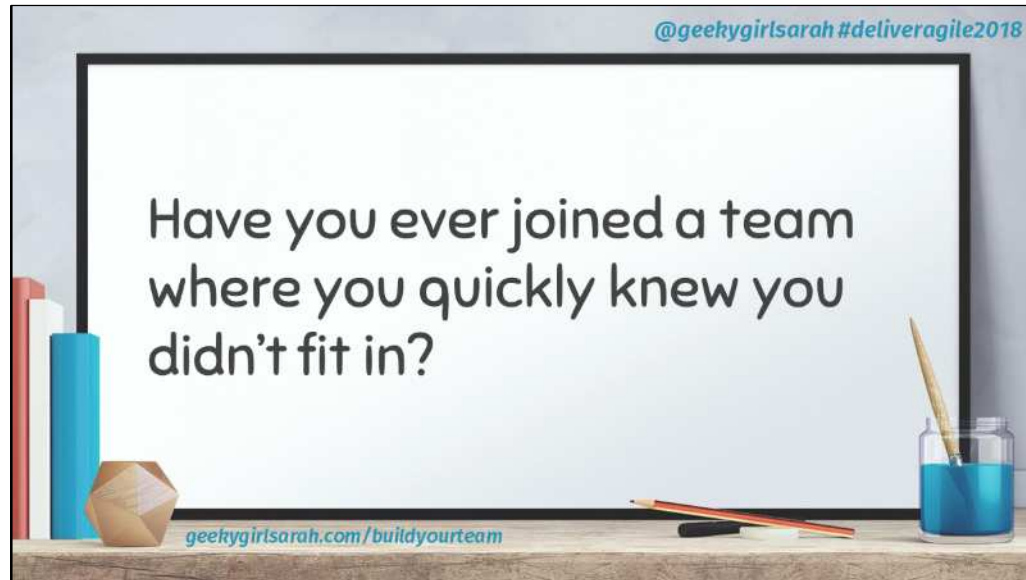
- Intro
- Definitely tweet along if you want



- [READ]



[READ]



[READ]



[READ]

OUTLINE

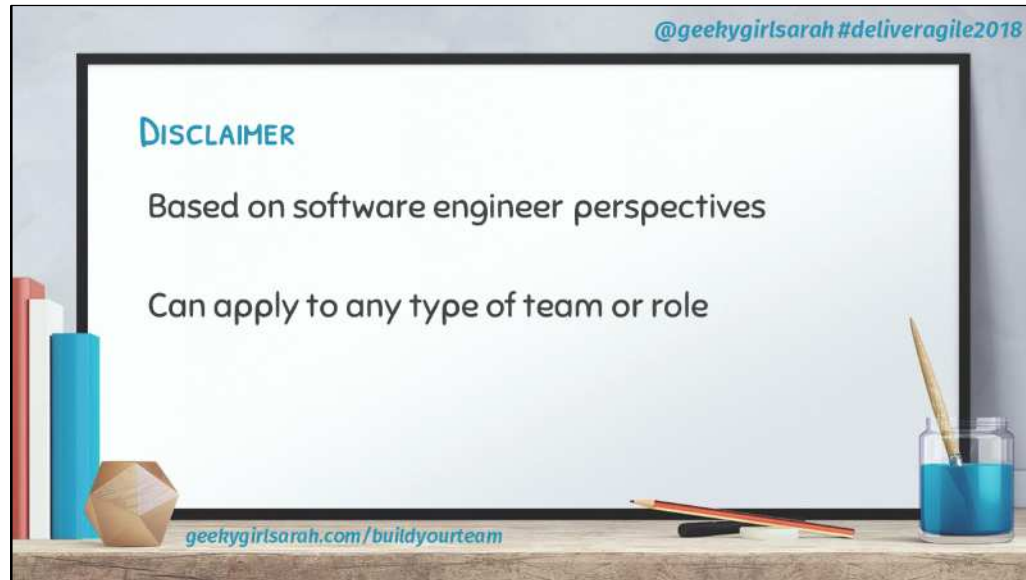
1. Intro
2. Choosing Good Team Members
3. Great Onboarding Experiences
4. Effective Mentoring Practices
5. Iterate and Improve
6. Conclusion

[READ]

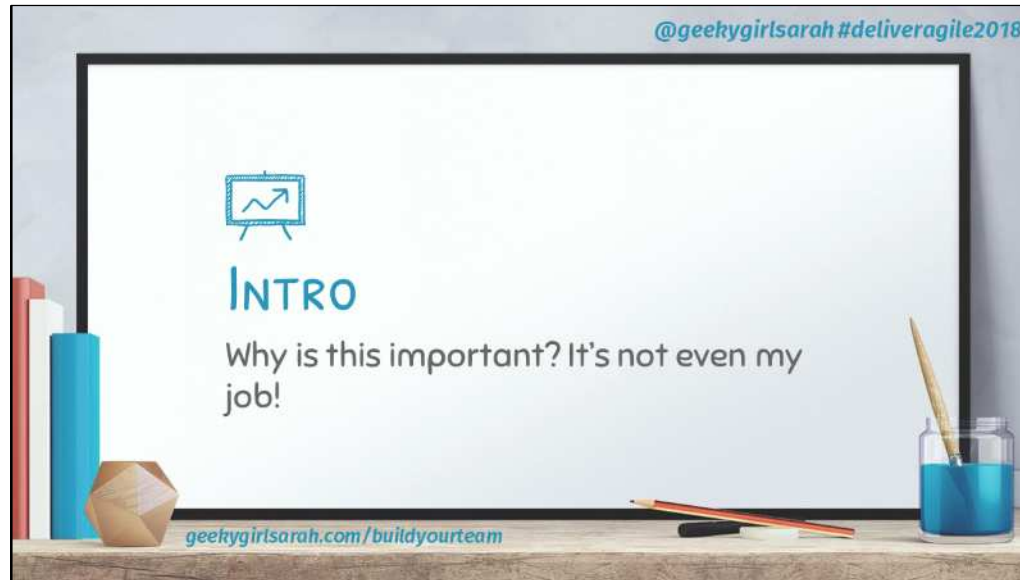
Who?

- + Developers/Designer/Writers/other team members
- + Managers/Scrummasters/etc.
- + Junior-level people
- + Senior-level people
- + ... basically any part of a team

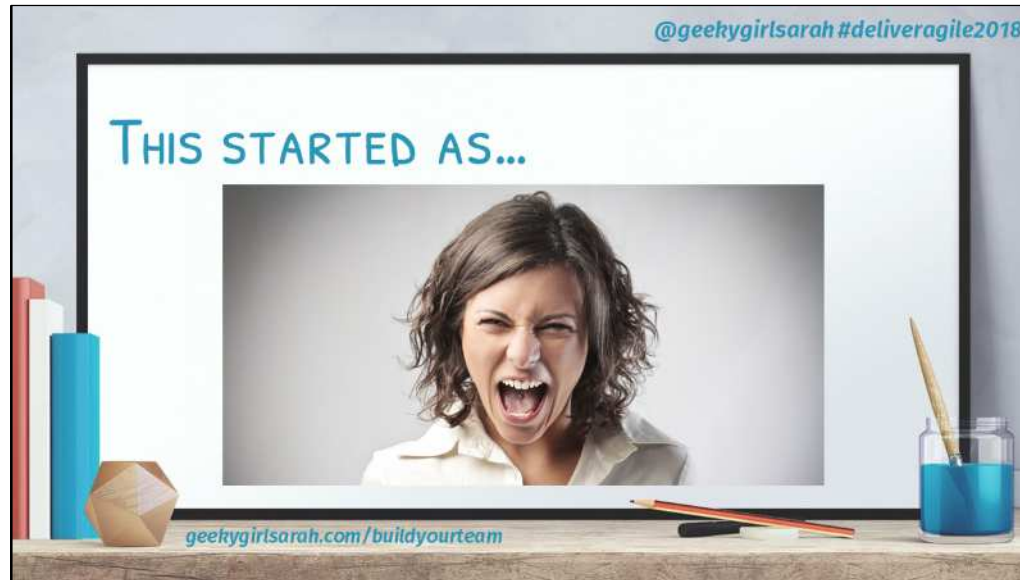
[READ]



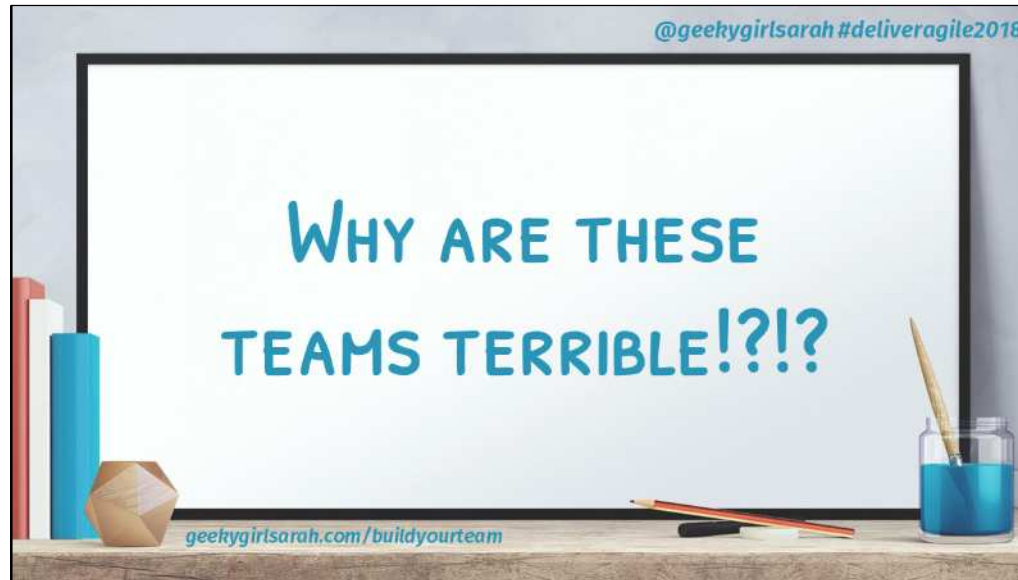
[READ]



- You might possibly wonder why you should worry about all of this stuff.
- You might not be a manager
- Or in HR
- Just a lowly grunt low on the hierarchy



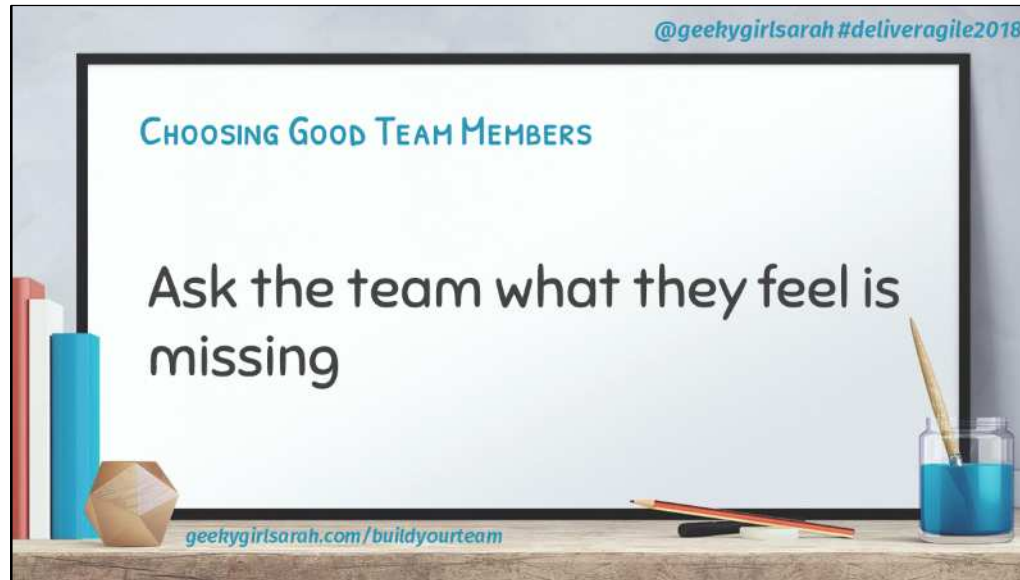
- This started out as a rant
- Talked to too many slack and Twitter friends
- SO MANY terrible stories of their work places
- It drove me mad



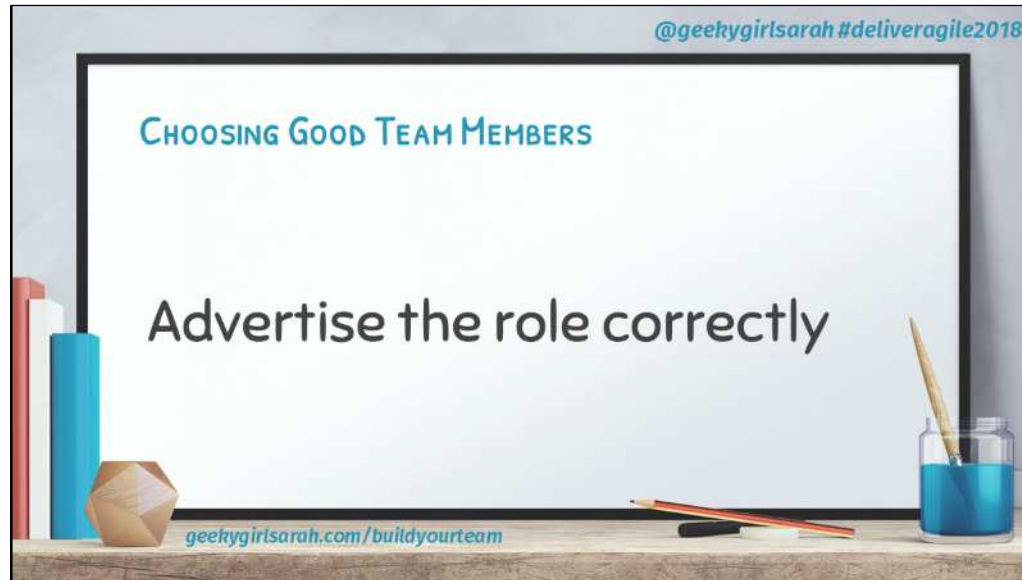
- Why are these teams terrible?
- What can we do about it?
- I proposed this talk to a conference couple of years ago
- Since then, I've learned even more about teams and what works as I've went through 8 month job hunt
- Also learned more from being on new teams and hearing even more terrible stories



- The process of building a good team starts before you even try to build it
- There's many things to think through before trying to hire a new person or fill a gap that opened up



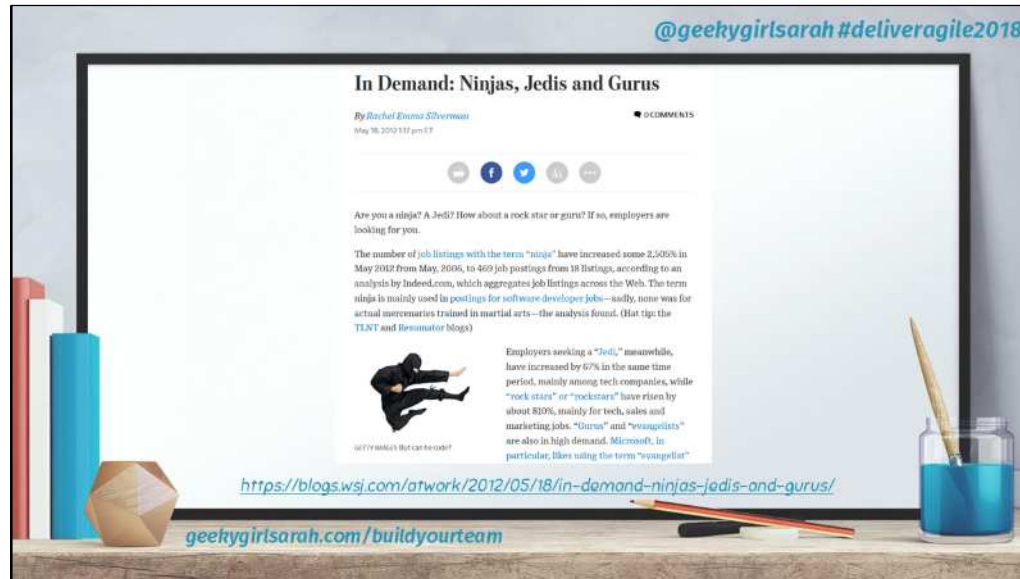
- Sometimes it's obvious... someone left, you need to fill the gap
- Sometimes it's not... you need a new team member
- What tech expertise do you need?
- What skillsets do you feel would be helpful?
- But also: What traits do you feel you need? A good experienced architect? A creative mind?
- Is there something you feel the team is lacking?



- [READ]
- And I emphasize “correctly”



- This is from an actual developer advocate role I was asked to apply to
- Description is:
 - Bachelors in computer science
 - Master's in management
 - 4 years of experience in being a dev advocate
 - 3 months of training required in software, marketing, application
 - Deep technical background with at least 5 years of software development before your 4 years of experience as a developer advocate
 - Outstanding public speaking abilities
 - Deep understanding of technology and marketing
 - Proven record of acceptance to major software conferences including [list of big name conferences]
- It's unreasonable for what the job turned out to be because after interviewing me, they considered me a fit. I meet like 1-2 of those
- Even worse, it wasn't what they even wanted in the end. They took this down later and replaced it with something different.
- I looked on there yesterday and the position is back up and it's a small fraction of the requirements
- THIS is preposterous. Be accurate! Be descriptive of



- This is an article from the Wall Street Journal in 2012. They said that:
 - From 2006 to 2012, jobs with "ninja" increased 2,505%
 - Jobs with "Jedi" increased 67%
 - Jobs with "rockstars" increased 810%
 - Sometimes there's mashups, like "badass rock star code ninjas"
- Are there people that might feel turned off by these terms?
- More important: do these really tell what the job is?
- Be as specific as you can. If it's a role that may swap teams or projects, you can be more generic. But still be truthful



- There's studies that show people are entirely biased on candidates based on name, gender, age, and so on
- While we want to keep some things in mind, you really want to judge them based on fair criteria
- Think this through and establish it before you start interviewing



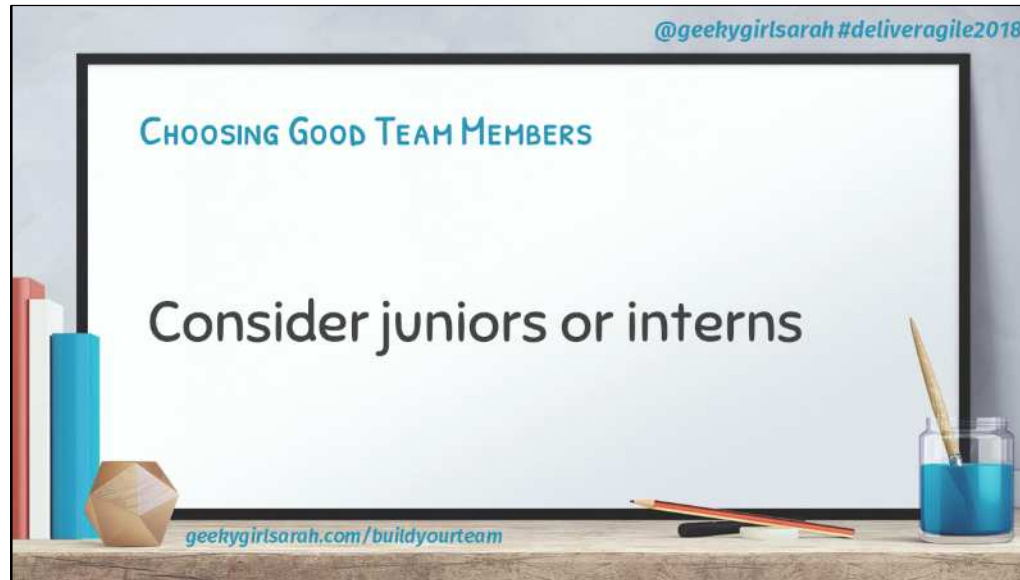
- Some of the best interviews I've ever had is where I talked to what would be my ACTUAL future manager or ACTUAL team members
- At New Relic, I flew to Portland and I had chats with some team members, pair programmed with others, and lunch with all of them
- At Square, I talked to the actual manager that was going to hire me
- At Arcadia I chatted with the whole team and paired with the manager
- My worst interviews were usually coding exercises that nothing to do with the job and with people I would have no involvement with later
- Getting a feel for your team is important. You aren't just doing work with them, but you're in meetings, you're collaborating. This is important!
- If you're not doing this, start pushing for it



- It's important to get a blend of diverse traits. Why? Studies show the more diverse teams build better products.
- Andrea [AN-DRAY-UH] talked about this in her keynote yesterday
- I've been in on meetings where everyone was senior developers and I was a junior, and I came up with solutions that were not only less complicated, but WAY more efficient



- You want a blend of things.
- You probably can't get ALL of these, especially with only 5 or so people
- But you CAN get some of them, and that will yield the most importance
- My team hired someone else and I at the same time.
 - I provided a really wide breadth of experience and would bring a lot of ideas and solutions to the table
 - The other new coworker provided a really deep experience in designing certain systems
 - Both of these were things this team needed
- Note what your team lacks on these traits, and keep that in mind when you hire someone
- Use it more as a final deciding factor than a "we need to hire a young person"



- It seems to be a thing I see a lot that companies and teams want the already experienced people
- That can be fine, but there's a TON of benefits to everyone by bringing on fresh talent
- First, some misconceptions

MISCONCEPTIONS OF JUNIORS

- + They are effective mind readers
- + They understand exactly what's expected of their code...
... as they walk in the door

geekygirlsarah.com/buildyourteam

- [READ]

MISCONCEPTIONS OF JUNIORS

- + They understand the nature of working on production code of a team
- + They'll get it right the first try

geekygirlsarah.com/buildyourteam

- [READ]

MISCONCEPTIONS OF JUNIORS

- + They're 20-year-olds fresh out of college/internships
- + They all have the same knowledge from their education

geekygirlsarah.com/buildyourteam

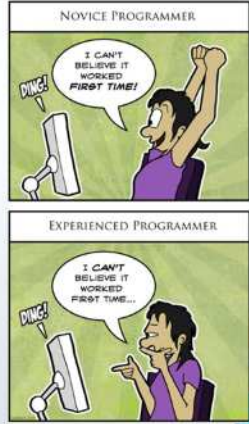
- [READ]

@geekygirlsarah #deliveragile2018

MISCONCEPTIONS (PARAPHRASED)

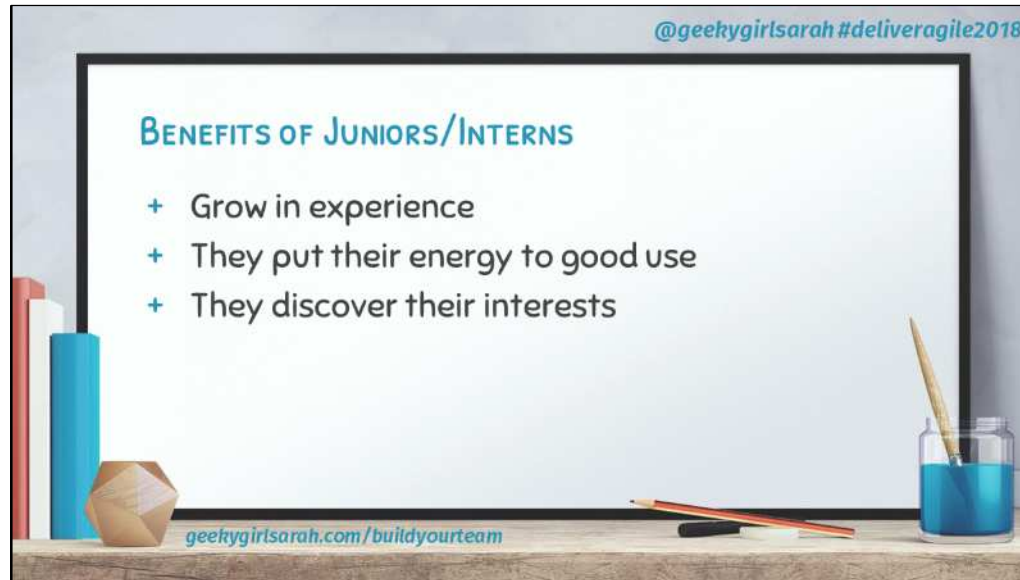
Lack of experience != Lack of ability

Being new != Being stupid

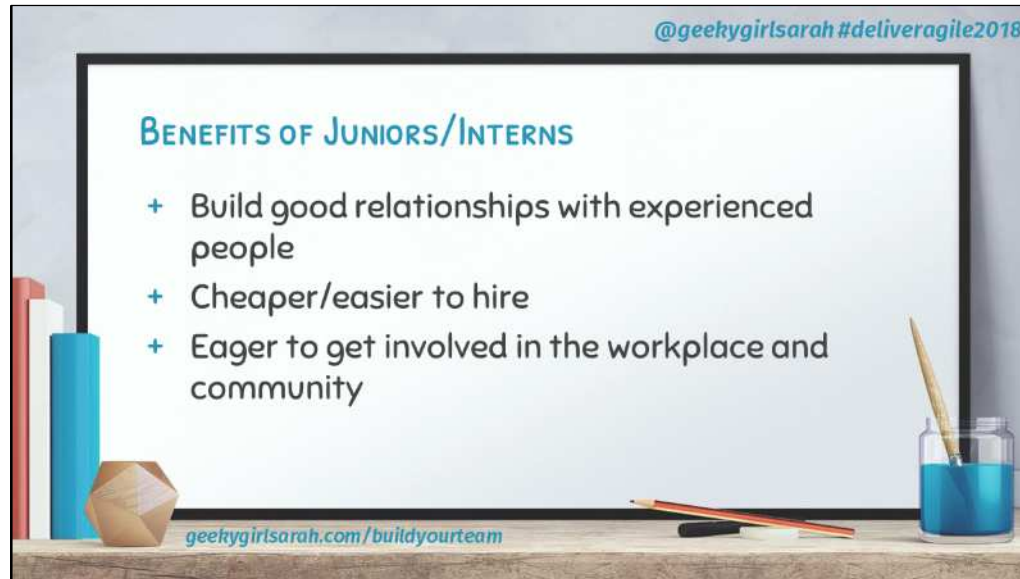


geekygirlsarah.com/buildyourteam

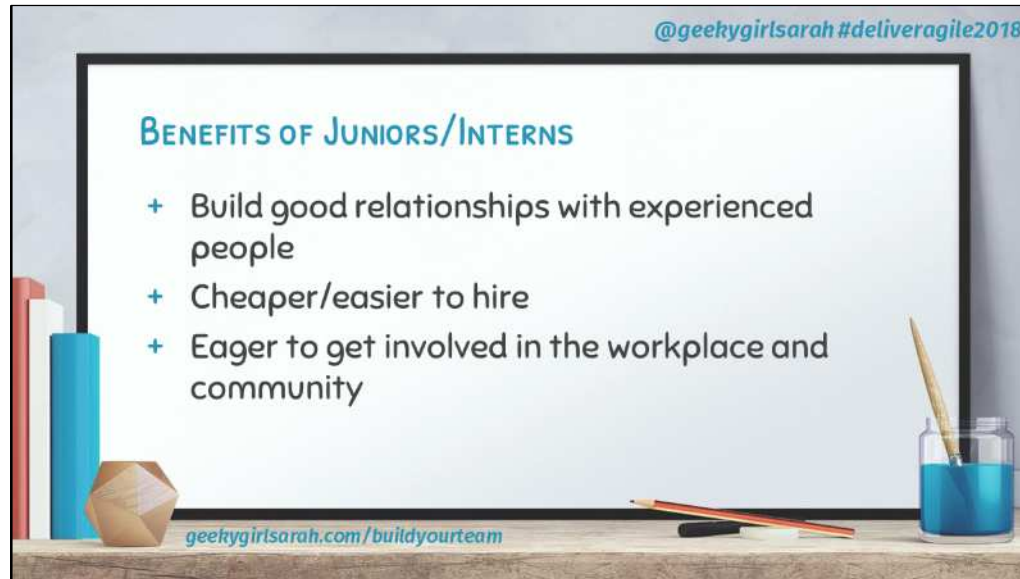
- This can all be paraphrased as:



- You learn WAY more from an internship than from school
- They're eager to learn and usually full of excitement. USE that energy!
- They get to discover what they are interested in. Did you know right after school what you liked to work on?



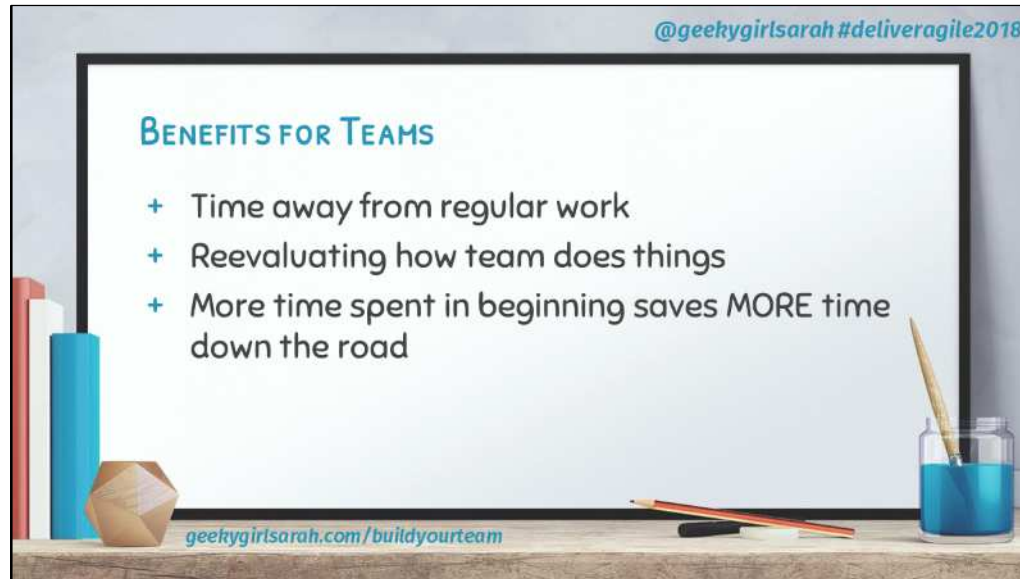
- Networking is valuable. It's great for them to meet more people, but also to see how experienced people work.
- This is usually why people want to hire them. They ARE cheaper and easier to hire.
- When was the last time you got another company outing email and you said "eh, not worth it." They usually want to do these things, which looks good for the team and company



- Networking is valuable. It's great for them to meet more people, but also to see how experienced people work.
- This is usually why people want to hire them. They ARE cheaper and easier to hire.
- When was the last time you got another company outing email and you said "eh, not worth it." They usually want to do these things, which looks good for the team and company



- [PAUSE]



- Seriously, how many times have you been so wrapped up in your project that you wanted a break? Helping them get started can be that break!
- Having a fresh face look at how things work is a perfect way to find out what terrible processes you have in place.
- It seems counterintuitive, but if you spend more time up front, they'll be more solid when they work on their own. You'll get more time later and not get interrupted as much



- It's not always easy to transition to new tech, but it's even harder when you don't know what it is. (Let's face it, it's hard to keep up with it all. Even here I've learned new things I didn't know how to do.)
- Ever feel you're in a rut? Fresh new ideas can help break that.
- It's hard to think about the team's future, but building an intern up early and giving them a good experience means they'll likely want to return to your company. And that's a SUPER easy hire. They already know what they're doing!
- Does your team have things they wanted to do but no real time to do? Give those projects to new people! They can help aid in productivity in ways you didn't expect.

CHOOSING GOOD TEAM MEMBERS

- + Ask the team what they feel is missing
- + Advertise the role correctly
- + Have standard and fair rubrics for comparing candidates
- + Get team involved in interviewing
- + Ask the right questions
- + Look for diversity
- + Consider juniors or interns

geekygirlsarah.com/buildyourteam

In summary...



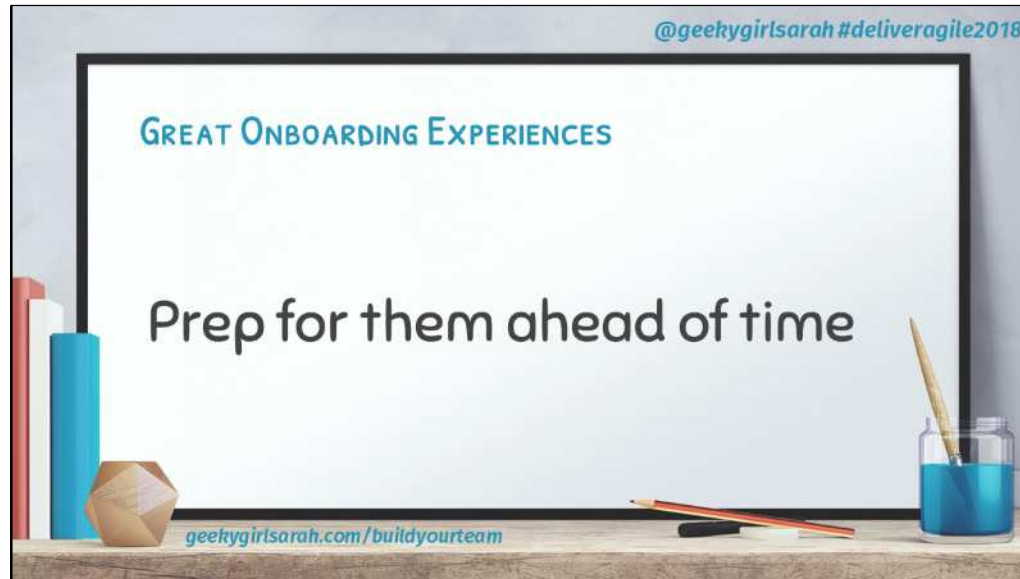
- With this, you can now enter into the next phase of this process...



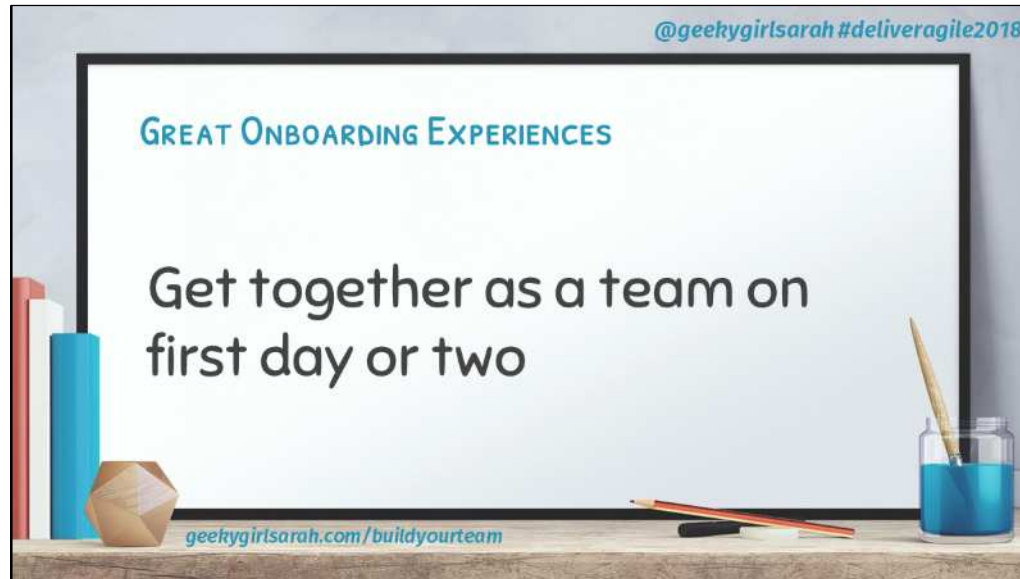
- You've hired someone! Now what?
- You have to bring them onboard. This can be good or bad, but usually it's rather... meh.
- It's usually some combination of good and bad from what I've seen
- ...



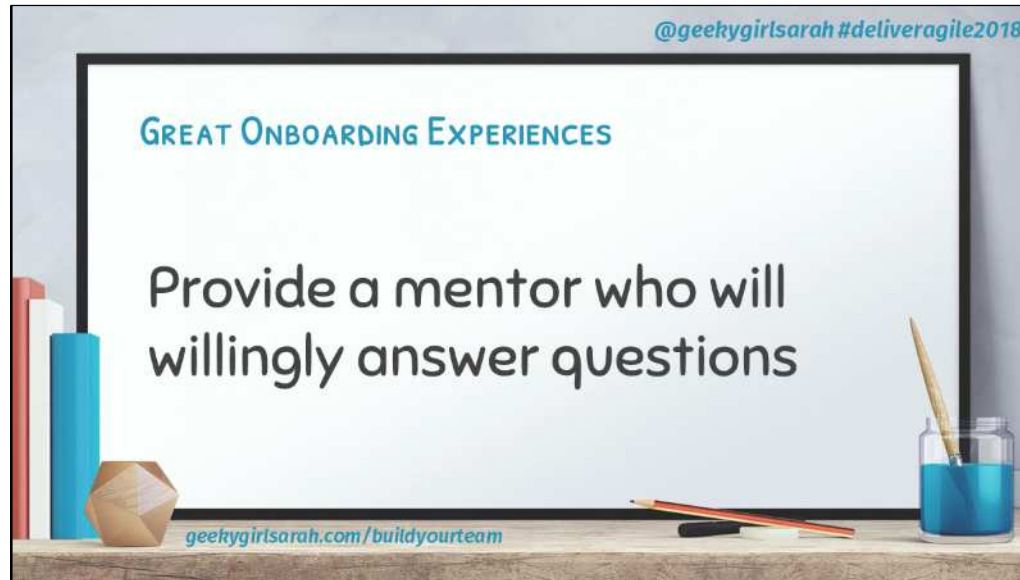
- Rarely is a process really good though.
- What can we do?



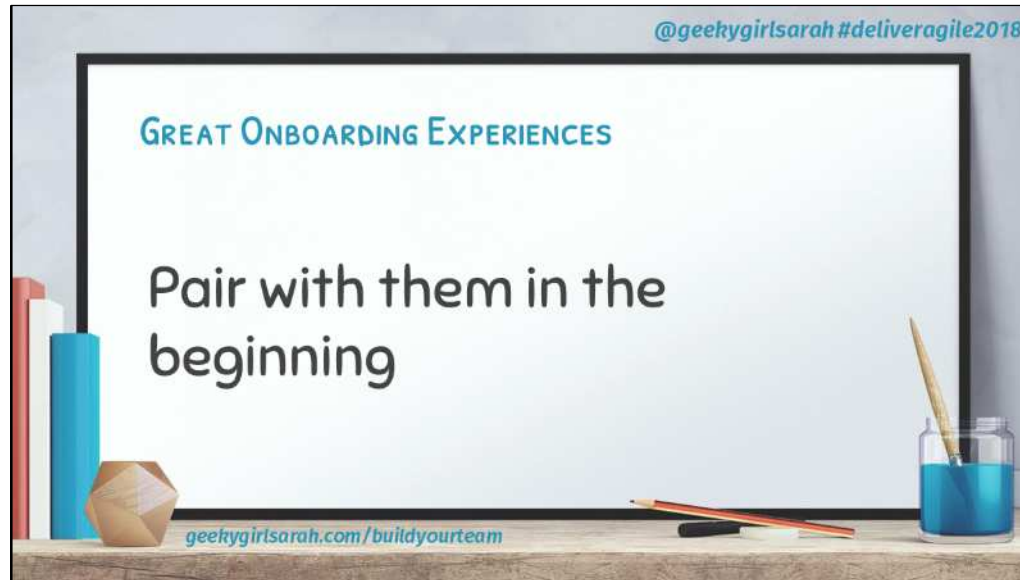
- You know they're coming... you're the team that hired them
- Get the computer mostly ready
 - If it needs software, either have it ready with keys if needed
 - Or have easy to access place with all the software
 - Maybe a github repo with a provisioning script to download/install it all
- Logins should be tested if possible
 - It's super annoying to have to wade through tech support the first day
- Have desk ready and clean
 - Seriously, I've moved into disgusting desks. Dusty, several year old can of coffee, etc.
- Besides being easy and convenient, it makes them feel welcome



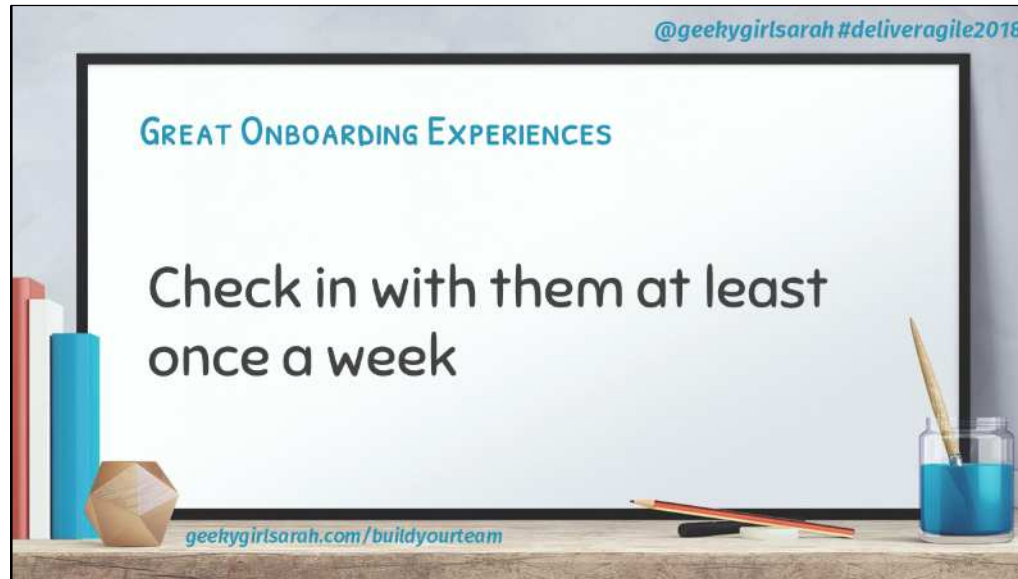
- ... and I don't mean to work
- Play a card game
- Go to lunch
- Just talk about backgrounds and where you came from
- Go get ice cream (I did that with a team once)
- This builds a good vibe with the team. If you care enough to spend time learning your new teammate, it reduces their stress, but also provides more of a safe feeling for them to be included
- Less like "token new person"
- (We all hate being the NEW person)



- This is huge. The “Ugh, who is going to mentor the new person? No one?” is not a vibe you want to have
- This person should be easily available (desk nearby, chat client), etc.
 - You don’t have to be instantaneously available (don’t want to break “the zone”) but they should have a way to get answers easily



- There's no better way to figure out how the team works than letting them see what you do. Then letting them do it too.
- If you already are a pairing team, spend some extra time with them in the beginning (don't trade pairs as often)
- You can even pair on non-code-related things
- If you're a mobbing team, maybe sit out with them for a little bit and show them how everything works, or let them watch the mob a bit first before diving in head first.



- This isn't a "how's it going? ... That's good. Have fun." kind of thing
- Actually check in.
 - How are you doing?
 - Are things making sense?
 - What can I explain better?
 - Are there things I can provide to help things make more sense?
- Often they don't know what questions to ask, and can feel overwhelmed quickly. So check in! Even experienced people need this too.
- There's a LOT to absorb in a new job. We know that.
- Ideally... more than one team member checks in too. (Team members can do it once, but new person gets more support)

GREAT ONBOARDING EXPERIENCES

- + Prep for them ahead of time
- + Get together as a team on first day or two
- + Provide a mentor who will willingly answer questions
- + Pair with them in the beginning
- + Check in with them at least once a week

geekygirlsarah.com/buildyourteam

In review



- Great! So they're set up more to succeed, and the team is getting ready to benefit more from their involvement. What do we need to do now?



- I debated putting this section in my first version of this talk. After a ton of managers and experienced devs told me “DO IT! I DON’T KNOW HOW TO MENTOR!” I knew just how important this was
- There’s three main things you need to know, and I’m going to dive into them.



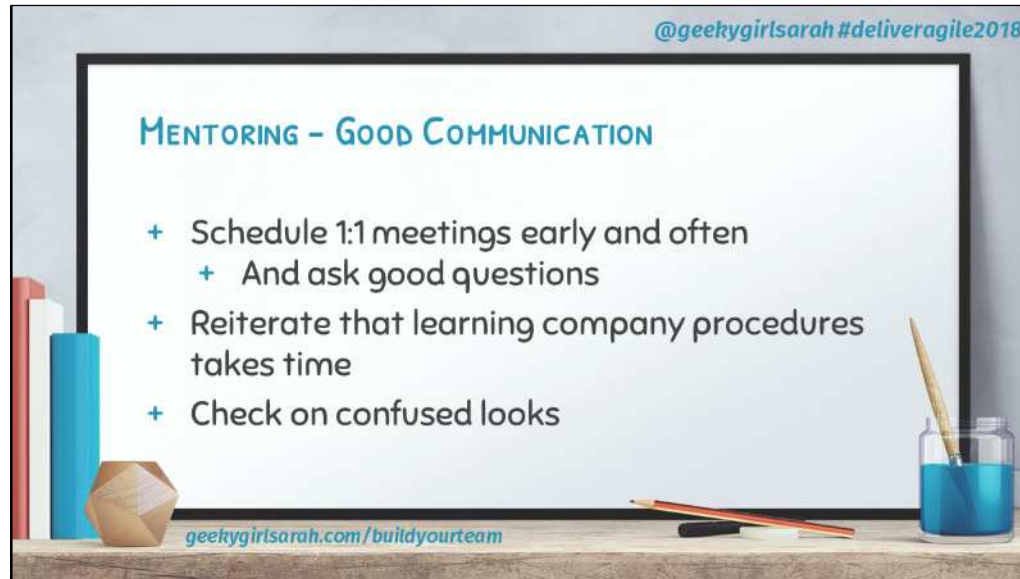
- Mentoring feels like two people sit at a table and magic happens. This isn't the case. It takes some thought and diligence.



- We talked about good onboarding. But a good onboarding is what helps start the mentoring process
- Standards for how you're looking at their work need to be consistent with what the other team does, but also fair.
 - Consistent in that you treat them the same way you treat other team members
 - Fair in that you don't have high expectations up front. Those build up over time
- New people almost never know what they need to know. "Do you have any questions?" almost always results in "no". So it's really 100% fine for them to literally not have a clue what to ask
- While you should guide them along in the beginning, leave room for them to learn things and try out stuff.
 - If you're pairing, let them fail sometimes and guide them to the right way to do it. (They learn by doing)
 - Give them time to play with your software, learn what it does, see other people use it, etc.



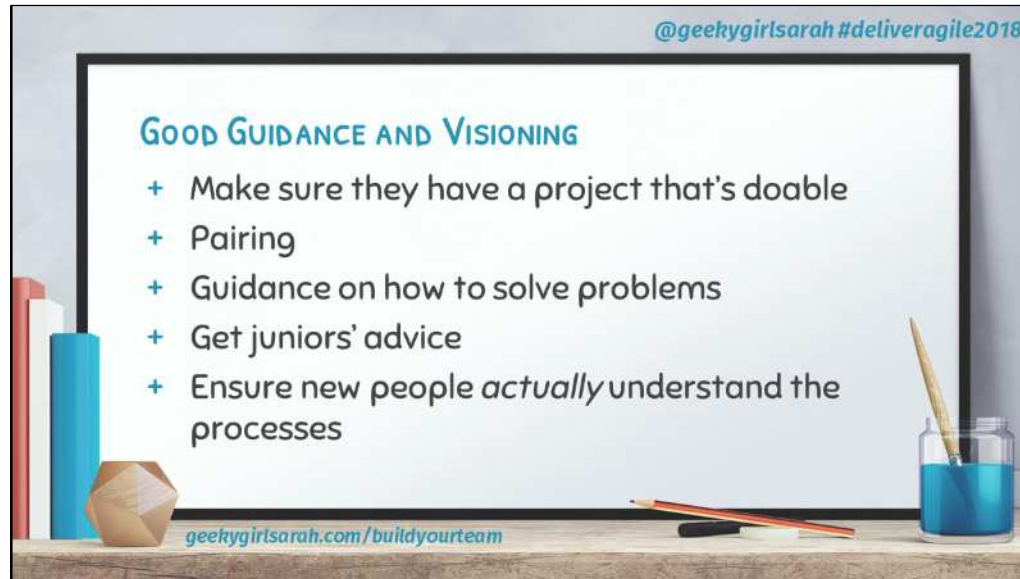
- I've heard so many talks here on communication being a major key to success. Same thing here. What does this look like?



- Don't wait a month or two to do 1:1 meetings. Start them early! I had one my first week of this job.
 - "How's it going?"
 - "What are you learning?"
 - "What's going well right now?"
 - "What's not going so well right now?"
 - "What can I or the team help you with to be better?"
- Remember that learning your company's processes takes time. And if they forget things, it's ok
 - I feel like I stumble awkwardly at every new job for a while
 - I also feel like things taught to me in orientation don't always hold true in actual work
- Check on confused looks
 - They don't mean "I understand". Ask what's wrong. Ask what you can re-explain.



- Mentoring really is about guidance. You want to help them learn what they need to do to be better and succeed. How do you start this off?



- They should always have something to do from day 1. (You may not get to it on day 1, but it should be there.)
 - My last job didn't give me a specific thing to work on for about 6 months
- Pairing. Like I said before, letting them SEE how things work, and giving them guided opportunities to try is the perfect hand-on experience
- Talk with them about how to solve problems. While they may have some experience on how they might approach something, they don't know your system well enough to usually implement it yet.
 - My last job also did this... the way I would try to solve problems ended up being "wrong" because they had implemented stuff in some part of their framework already, but it was hard to tell
- Get your less-experienced people to try to contribute on larger decisions
 - You'd be amazed at how well they could solve problems in totally different (and sometimes simpler) ways than you might
- Ensure they ACTUALLY know the processes. It's easy to hear a list of 10 things and say "yeah, I get it" then sit down to do them and remember only the 8th item to do.
 - Have them re-explain it
 - Have them write it down



- Mentoring really boils down to these 3 things. If you can focus on them, and keep them in mind, I think not only will the new person benefit, but they'll integrate into your team faster, and the whole team benefits from this.



- Mentoring is great! Pass it on!
- So you've brought on someone, you taught them the ropes, and they're supported. This is great!
- We need to do one more very, very important thing.



- I would be kicked out and have to go home early if I didn't incorporate something Agile into this...
- But doing all of the above work is really not beneficial to the future if you don't iterate and improve this process



- Your "sprint" (if you will) is probably the time period of start to finish.
 - Where start is pre-planning your hire (the first part of this talk)
 - And where end is 3-6 months into this (this part of the talk)
- Plan this out
 - I'd recommend documenting what you plan to do, so every part of the first part of this talk
- If you hire often, this may encompass more than one person at a time (ex: 5 people a week)



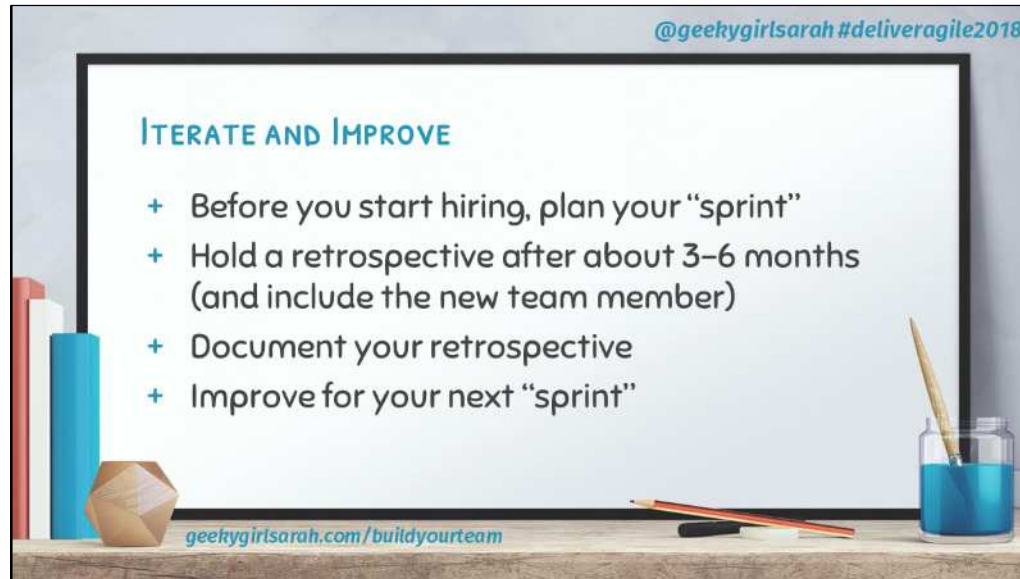
- A retro is going to tell you what went well, what didn't go well, what to keep doing, and what not to keep doing.
- As a team, you can probably see what happened if it was a rough process, or see how well it worked if you had great candidates and you made a good choice
- And INCLUDE the new person! They had to endure the process, they can tell you what to do to make it less exhausting, easier on them, and maybe some other things
 - This worked at my last job. When I finally told them I was job hunting, they revamped their interview process to find someone that would be a better fit and not with incorrect assumptions



- You probably document your regular team retros... do it here too!
- Why is this important?
 - It helps you remember what to do next time (if you don't hire often)
 - It helps other teams get ideas on how to improve
 - If you do hire often, your process is probably more complicated. Documenting will help everyone in the process see what to do to improve.



- Every interview can be better.
- Every onboarding can be better
- Every mentoring experience can be better
- Use what you learned to improve on this! Why shouldn't you make it better for everyone involved?



- This should more or less look like what you do every 1-4 weeks anyway. Do this for your hirings and onboardings. I think you'll see it makes a difference.



- As my friend Allison says, "Team work makes the dream work!"



[READ]



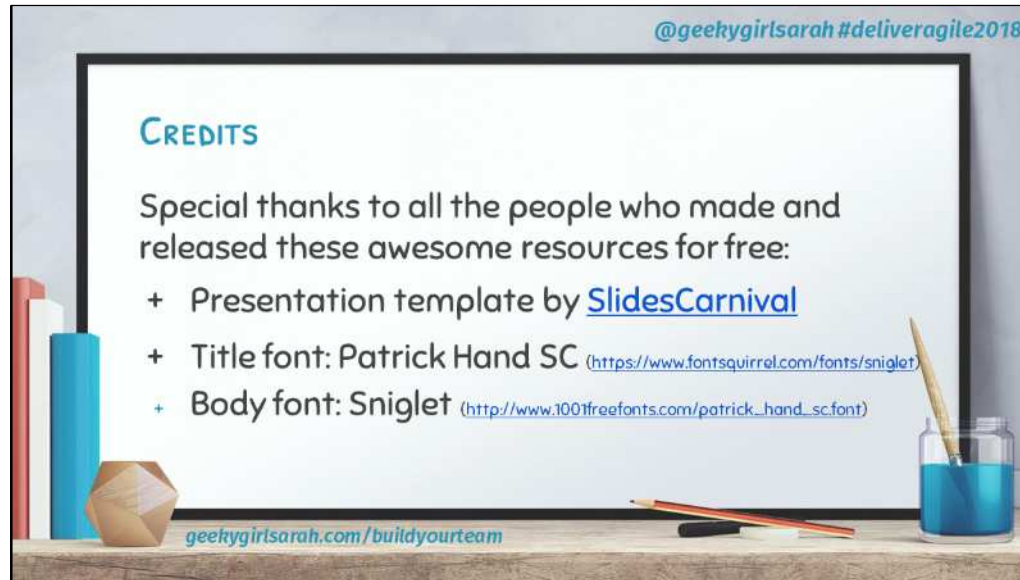
- These are the 4 main things you need to keep in mind.
- These things will not only build your team, but build it to last.
- The new people will benefit immensely from this help
- The team will grow and be a better and more productive in the long run
- Your company will get better at this process as you keep doing it, whether you're a small company or a large one
- It's a lot of stuff I went over, but you know what?



- YOU GOT THIS!



- I'm Sarah Withee
- Geekygirlsarah on social media
- I'm happy to chat afterward up front or in the hallway
- Go! Be awesome!
- Thank you!
- [STOP]



Reference slide (not shown)



Reference slide (not shown)