Makers Portfolio - Lauren Qurashi: March 2019

Table of Contents

I Can Make Anything

I Can TDD Anything

What does it mean to be able to do this?

The art of Test Driven Development is the practice of using tests and error messages to build your code. The process generally goes:

- Plan Plan Plan! - in any way that's helpful to you, make a plan of how you'd expect your

software to operate. Remember, this plan will change constantly, so don't get caught up on it.

- Run a feature test - pretend to be the user and run a few commands you'd expect to be able to do. ie plane.takeOff or click_button 'Edit'…

- Use the error message - your computer will tell you what the issue is, in this case the plane won't exist.

- Make a unit test - The best idea with this, would be to make the smallest test going, for example, check that your plane class exists.

- Make both these tests pass - by using your error messages, googling, previous knowledge, a friend's help, do what you can to write the code that passes this test in the smallest way.

- Plan a new feature - now you have your plane class… use the process to make it respond to the fly method.

  To me, being able to TDD anything means that you can approach any type of software, and use any type of TDD'ing technique to further its development. This could include your general code, or even just problems in everyday life, all in an attempt to understand how things work. Good TDD practice includes starting with small tests and doing the absolute least to be able to pass them, all while ensuring you don't assume anything about the code you've written. To combat any assumptions, you should also test edge cases and the existence of your classes / objects. Good TDD ability includes aiming to test behaviour (not state), and applying processes such as the Red to Green Race technique. Another good practice is committing your tests to Github before deleting them once rendered obsolete. While I may be no way near close to knowing everything there is to know about coding and problem solving, I believe the ability to TDD anything comes from the practice of applying this method to driving development in whatever you may be focusing on.

Evidence and explanation

- Me TDD'ing Fizzbuzz Javascript

- The Notes App - Javascript Repo. This project was our first introduction into testing without a testing framework such as RSpec or Jasmine to help us along. I feel that this repo shows that I can write clear, concise tests, without the help of a testing framework. The tests in this repo follow 'the next smallest step' kind of fashion, and are also riddled with doubles and mocks to ensure that I'm testing what I want to test. This repo also

shows a good example, in the notes-list-spec.js file, of my fave TDD'ing process Red to Green. I started off testing the smallest thing that is required of this object - that my notes list contained an array. The tests progressively grow in small steps, into a list of notes that I was completely confident of. This testing checks the behaviour of the list rather than the state, and ensures I am testing the right thing by employing doubles. In the first 5 weeks, I really struggled with mocks and doubles, especially in Rspec. I found I could completely understand their purpose and how they worked, but I just couldn't implement them into my work. After moving onto javascript, I found I could easily implement doubles into my work which flourished my ability to identify the need for them in my projects. I now feel comfortable in saying that I have a good understanding of not only why we test, but how to test effectively.

- Airport - Ruby repo. This repo shows my first experience in TDD, and while it may not be the best example of testing, it helps illustrate how far I've come since week 1. I do feel however, that this example shows that during week 1 that I got stuck into the testing and shows evidence of me understanding the basics of what a good test is.

- RPS challenge - Ruby Repo. This is only a small example, however I wanted to also express my understanding in feature testing. I never really got far enough in most of my projects to start implementing intense CSS and other features, however the RPS challenge has a few small examples of my understanding of Capybara testing.

Feedback

Sam: Makers BnB - “You have a solid understanding of TDD even if you can't always get the syntax to work for you. The syntax will come (and will change every time you change language) but you clearly understand and strive for following the process of writing a test, failing a test, passing the test. This helps you to write only what is required without jumping ahead.

Lisa: Mentee - "Thank you for showing me the ropes with TDD, I was struggling a bit with it at the beginning but you've made it a lot clearer, especially the feature tests. It's hard to understand when there's nothing to see with no front end. I was even able to explain the doubles thing to someone else on my cohort so I'm pretty sure I get it now."

I Can Program Fluently

What does it mean to be able to do this?

Being able to program fluently is a tricky question, however I feel that it means the basics of initial programming comes naturally to you, and you can execute this with good

practice. For example, if faced with a problem to solve, someone who can programme fluently would instinctively start planning potential solutions, including writing up examples of psudo-code, in a vague idea to what would be required for their user stories. Following that, a fluent programmer would be able to translate their psudo-code into actual code with a little research here or there. The core of being a fluent programmer (to me at least), means being able to do all of this while following proper convention, and producing clear and concise code.

Evidence and explanation

- Repos showing good quality code - confirmed by coaches. (Bowling? Instagram repo?)
- Screen Recording of me doing a process workshop.
- Instagram Repo? (if you get a chance)

Feedback

Paul: Oystercard - "I was really impressed how you managed to jot down a load of potential options on a plan for how your station - card interaction should work."

Kat (coach): Battle - " You've done a really good job of making up this view file, I can see CSS and HTML is something you have a knack for!"

I Can Debug Anything

What does it mean to be able to do this?

Being able to debug anything is a bit of a big statement to me, however I feel that to be able to debug anything means being able to approach any software problem you may face, and attempt to tackle it by having a methodological approach to finding and solving a bug. Debugging anything does not mean that you're able to solve all the problems you'll ever face single-handedly, but I feel that having the confidence to follow the process and come to a verdict of what the issue is, makes you able to debug anything, as all that follows from there is learning how to fix the bug.

My method for debugging is a simple (and very common) one:

- I'll start off by tightening the loop with my error message, making sure I've completely read it, especially the type of error and the line it occurs on. You're not always lucky enough to have this, especially if you don't have a framework. So make sure you take

advantage of your stack traces and your error types.

- The second step is to go to the line of conflict and investigate. The best way to do this is to get visibility of all your variables, methods, etc. So print out what you can to fully understand if you can see where something is going wrong. If you find it, great! if not, is there anything that might be having an influence on it like another method?

- The third step is to solve the bug. Once you know what's going on, only change one thing at a time. You don't want to be changing everything and not know how you fixed it, or even worse, break it more. Solving the bug can come in many forms. My first thought "would be can I solve this?" second would be "have I seen this before?" after that, it would be to go and find out. I'd google the error message, the type of problem I'm having, and should that not be fruitful, asking for help is never a bad idea.

  I always used to have a bit of a complex when solving bugs, I used to think that if I couldn't do it, it was cheating to ask for help. But never suffer in silence, it's not productive. One big thing I've learnt is to not be so hard on myself, this stuff comes with knowledge, and that comes from experience. If you want a challenge but need some help, try just asking for a hint at what to look at. But there's nothing wrong with learning from the actions of others.

Evidence and explanation

- Me Debugging Acebook

- Screen recording of Acebook debugging - The bug that we squashed was lurking round the create method for new instant messages. Any time we'd try to create a new message, It would throw a NoMethodError. I started off by reading the error message on the screen, to find out what area was giving us grief. The error said that it was in the messages HTML page, so I went to investigate. There was nothing in there for me to really get any visibility on, so we moved back a step to the controller as I knew the controller needed to deal with any variables that the views needed. Once I had found an area to investigate, my next step was to get some visibility. I printed out the message instance variable to see whether it was picking up on what a message was. The stack trace in the terminal showed us an empty message instance so I knew that it wasn't an issue with the message itself. The error message we got in terminal showed that 'username' was an undefined method for user. This prompted me to look at what features I could call on my user. If it wasn't for this, my next option would have been to investigate whether my messages instance was holding the all the messages properly. Once going through the database for the Users, saw that I didn't actually have a username for my users, rather a first and last name. This prompted me to change the username to first name, which solved the bug and printed out the name of the user typing the message.

Feedback

Bart: Makers BnB - “I was also impressed with your Github best practices when debugging. You were always conscious of making sure that we tested one thing at a time, and always following the correct protocol.”

Jeremy: Battle - "you're pretty good at making sure you know exactly what you're looking at. I never would have thought to check the things you were looking at to see if they were what you expected."

I Can Model Anything

What does it mean to be able to do this?

Being able to model anything means being able to get your ideas down on paper as to how a program should flow. Modelling will always happen multiple times in the development of software, but if you’re able to clearly note down your ideas about what you’re working on, then you should be able to fulfil these criteria. I also feel that an important part of being able to model anything includes the ability to do such a task in a way that is clear to others who may not be technologically inclined.

Evidence and explanation

- My Diagrams - I have a particular love for clarity, and while I can note down a messy plan should I need to, I much prefer to use Draw.io to make cute diagrams. I used to really struggle with modelling in the first couple of weeks, and I think a big part of it was I couldn't understand what the class diagrams from the workshops meant. After Sophie enlightened us to the various types of diagramming, I was kickstarted an awful lot after seeing that there's a wide variety of professional diagramming techniques. Once Kat told me that the most important thing is getting your ideas down on paper, I was much happier too. I've got a big collection of diagrams, from class to active flow, proving that I could diagram in a professional and clear way. However that's not as far as my diagramming skills go. In one particular workshop in week 3, a couple of my cohort members were struggling to understand the MVC model, and the HTTP Response diagram. Alice quite liked my ability to explain my point in a less technological way, so she asked me to make a poster to put up in the toilets which is pretty cool if you ask me. You can see it here..

  How the Internet Works poster for Alice:
  

Other types of Diagrams :

Feedback

Bart: Makers BnB- “It was really difficult to distinguish who did what during the week with the pairing sessions so feel that I can't really make an assessment from that. However, when we did pair up I felt the planning process was exceptional. The discussing, then researching, then implementing was top notch. I felt we bounced off each other well.”

Alice (coach): Poster - "Your diagramming skills are really good. I like the range of examples you've got and I like how you are able to put it into non technical and funny terms for others who may not understand."

I Can Refactor Anything

What does it mean to be able to do this?

This simply means that you have the ability to reconsider the code you have, in order to make it clearer. The code should be easy to read, and have a clear direction. Refactoring doesn’t necessarily mean making your code shorter, or more complex. Refactoring is all about making sure your code is to the best standard it can be, whilst being clear to others and easy to change.

Evidence and explanation

- Commit histories that show my refactoring.
- Showing that I’ve gone back over old repos and refactored my code.

Feedback

Matt T: Boris Bikes - "I really liked how you made it work first, committed it, then went back to refactor. It shows that you value functioning code before trying to change anything. I admired that when you refactored the Docking Station, you'd only change

something if it was still clear what it would do."

Alex: Acebook - " You're good at recognising what needed to be done in rails, and then seeing if there was another way of putting it, for example, the routing. You changed it from arrow notation to something a bit shorter."

I Have A Methodical Approach To Solving Problems

What does it mean to be able to do this?

A methodological approach to solving problems includes following a procedure in order to tackle a problem. More generally, it means having a thoughtful effort towards investigating problems, rather than trying things and hoping they work.

Evidence and explanation

My process to solving a problem is a short one, however it applies to nearly all situations in life that require a bit of research to get over an issue. It's quite similar to my debugging process actually.

- First I ask myself whether I understand the problem at hand. I need to be able to clearly explain the issue to someone else in order to move on.

- Once I know the issue, I ask myself if I know how to solve it. If I do then that's great, all sorted, if not, my next thought is to whether I've encountered it before. There's no point re-finding out a solution if I've got the code to fix it at hand.

- If both of those options fail, It's on to some trusty research to see if people on the internet can sort it for me through the medium of education. I'll usually google it, and if that falls through then I'll ask for help, there's no shame in that especially if you can get some useful knowledge from it.

- Then it's on to implementing the solution. If this doesn't work, it's back to the research step.

- Hopefully if I've managed to fix it then all is well! If I'm in a pair, then I ALWAYS make sure to explain what happened to the other person, it may be annoying / obvious to them, but it helps me consolidate what I did.

  One example of my approach stemmed from the week 6 makers BnB challenge. I put myself forward to work on making a SPA that interacted with an API we had created. The

main reason I had put myself forward for this task was due to never working with this sort of technology myself before, so to a large extent, the whole task was a problem to be solved. I still followed a process towards combatting this task however, which followed the guidelines of my approach. I understood that the problem was that I didn't know how to start off a Javascript file that dealt with JQuery. I didn't know what was included, or what needed to be separated into a separate file or pretty much anything, so that was the first 3 steps wiped out! I decided to research into the JQuery thing, and got an awful lot of good info, but was still stuck on what needed to be in a separate file etc. I decided to ask my team for help, who helped me piece together the layout to my page. I then explained what had happened back to my helpful team members, who confirmed I knew what was going on!

Feedback

Henry: Makers BnB - "I admired that you knew what to do when you didn't know what to do, in the sense that while you didn't understand the javascript file, you had an action plan of how you were going to find out. It shows it's helpful to have a plan with problems and bugs."

I Help My Teams Succeed

I Use An Agile Product Development Process

What does it mean to be able to do this?

working in an agile environment basically means to adopt a working approach that offers maximum flexibility, with minimal constraints in order to allow people to do their best work. This involves heightened interaction between all aspects of the project team to ensure clarity and harmonious synergy. In a more simple scenario, it means regular stand ups and retros to ensure everyone's on the same page, and being flexible with working hours and what best suits peoples needs. Basically a flexible working environment that's main goal is doing the task well.

Evidence and explanation

- Code Review of Acebook Merge Request - To follow an Agile methodology in our Acebook teams, we ensured that at least 2 people reviewed someone else's merge request. We added comments and sought after answers before confirming the merges. Along with Travis CI running, this ensured that everything in our master branch was good

quality code that could be deployed.

- Trello Board for Acebook - To follow an Agile methodology in our Acebook group, we also used a card board so that communications could be visualised by all. We assigned ourselves to tickets and used Fibonacci numbers to rate the complexity of tasks. This ensured that everyone knew who was doing what, and how complex / high priority of a task something was.

  In MakersBnB, We adopted a very Agile approach, as one of my team members had previous experience working Agile. We had stand ups every morning to check where we're all at and how we were feeling, along with retros every evening to see where we had left the project for the day. We also regularly checked in if we were struggling with something, and if it was a big problem we'd all mob it to sort it out. Agile working was very helpful for our group, as we worked towards the goal of strengthening our weaknesses rather than separating our roles out into strict departments.

Feedback

Michael (coach): Acebook - "I like the Fibonacci numbers on your Trello board, that's a cool idea. It also seems like you've got everything organised clearly on there."

Masha: Acebook - "Lauren did a great job ensuring we all knew what she was doing and was first in line to help if someone was stuck on something. It was also her idea to see if the group would be happy with independent study in the mornings to research stuff we're implementing in the project."

I Write Code That Is Easy To Change

What does it mean to be able to do this?

In the professional development world, you're going to have to anticipate changes to clients needs. They may decide a week before release that they want a completely different feature or a new skin on their UI. Being able to have code that can be easy to change is a vital component to fluid development. Having code that's easy to change, at the very base of it, means that another developer can look at it, and amend it into something different, without the whole project blowing up (if done properly). Things you can do to ensure this are:

- Ensure that your architecture is clear and easy to read. As I've experienced with rails, there's hundreds of files in a small project, so it's vital that naming conventions are followed and things are in the right place.

- Keep your code clean. Make sure all conventions are followed and everything is neat. Code should be easy to read and if it isn't then someone else is going to have a real hard time changing that should they have to.

- Make sure nothing is coupled. If making a change in one module / class / view is going to ruin everything then it's not very easy to change. Each class should have one responsibility, and classes should have as little interaction with each other as possible. The more you stick to these rules, the easier your code will be to change.

Evidence and explanation

- Talk about Oystercard / Boris bikes and how others would have to pick up my code.

- Talk about swapping in MakersBnB and Acebook.

Feedback

I Can Justify The Way I Work In a Business Context

What does it mean to be able to do this?

To be able to justify the way you work in a business context means that you can make decisions for yourself and explain professionally to your peers why you chose / feel your decision was required. This isn't just doing what you think is best, implementing it, and hoping that the others agree. This is putting forward your ideas and rationally explaining your choices with empirical evidence. If there was ever a time where you needed to make an executive decision and didn't have time for others' input, you'd have to be able to explain why you did what you did, and why it's best for the business.

Evidence and explanation

- Action Cable - Acebook - During the Acebook challenge, My team and I were set with the task of adding an instant chatroom to our project. After some independent research, we all came together to discuss our options. We found that there was a gem we could install that basically did it all for us. All of us agreed that the Gem would be the best option for our project, however I put forward the suggestion that we don't go down the gem route as it would be detrimental to our learning. I explained that it would be better for all of us to manually follow the Action Cable route so that we can understand what's going on. After hearing feedback from the group, they all agreed despite it elongating the implementation. Another thing we all agreed on was to take a but of a turn from the traditional Agile way of

working. Instead of giving individual members or pairs certain tasks, we all paired up and completed the same task in order to make sure that everyone's had a chance to practice implementing this.

- During the MakersBnB challenge, we also decided to work a little differently than the norm by playing to our weaknesses rather than our strengths. I felt it would be more beneficial for myself to work on things I've not dealt with before, and after speaking to the team about whether they'd be happy with this, they agreed and decided to adopt the approach for the whole team. We all walked away with some new knowledge and we even managed to build an API completely off our own backs and from our own learning. It's one of the things I'm most proud of during my time here!

Feedback

Masha: Acebook - "I'm pleased you mentioned doing things the long way with the chat messages as I felt like I was learning nothing with rails. At least I can walk away from this week with something new."

Sam: MakersBnB - "The way we played to our weaknesses was definitely a good move for us. Although our end product was significantly less developed than the others', I think the decisions you put forward helped us develop as a team and learn something new."

Sophie (coach): MakersBnB - "It's really good that you're looking at the two routes to take your project down. There's definitely an easy way and a harder way but if you think it'll be beneficial to you to take the hard way then go for it."

I Can Grow Collaboratively

What does it mean to be able to do this?

To be able to grow collaboratively basically means you can grow together, and in a team context, it means that you can expand your knowledge and skills by expanding someone else's. This could be in the form of teaching more junior members about topics you're knowledgable or just generally being cooperative with your skills. People who don't grow collaboratively tend to be those who adopt the ideology that sharing knowledge is lessening your advantages above others. Good developers work well with others to get everyone up to scratch, which will make your shared project goal a lot easier to obtain. I personally find it great to collaborate with people, I can leech all their knowledge about something, use it to make myself a more skilled developer, then help others do the same.

Evidence and explanation

- Before this course even started, I grew collaboratively an awful lot, more so on the absorbing end of the scale rather than the giving. During my pre makers life, I'd often pair after work with my developer friends on their problems to practice my pairing skills. I'd grown so much by listening and practicing that I grew strong enough to join the course. I'd paired with Junior front end Devs and Heads of Software in order to obtain as much knowledge as I could. I was then able to share what I knew with the cohort when I joined. An example of this is my knowledge of bootstrap, I was able to teach a fair few of my peers a much easier way of making pretty web pages by importing CSS libraries.

- During MakersBnB, my team and I grew together massively, not only in an emotional bond, but we learnt ridiculous amounts about things we dreamt up ourselves, all by working together. I developed my feedback skills as well as my ability to work with others who have a completely different style of coding to me. I came out feeling like I've grown as a developer, and have the feedback to prove it.

- When Alice overheard me explaining to my cohort members how the request-response cycle worked, she felt I did it in such a funny and clear way, she asked me to make posters to help educate everyone in Makers. This has been shown earlier in the portfolio, however I am proud of my abilities to work with Alice to make something that will help others grow their skills. It was cool.

Feedback

Harry: Battle - "I'm really grateful that you showed me how to use CSS, as I find it hard to concentrate for a while on backend stuff. Now I'm able to take a break and make things look cool which is a lot more satisfying."

Sam: MakersBnB - "You were really good at listening and being patient when pairing with team mates who were almost complete opposites to you, as well as developing yourself following feedback you were given. I noticed that when Bart mentioned you could be loud during pairing sometimes, that you were a lot more conscious of it and ensured you kept yourself in check to not disturb others."

Puyan: Developer Pal - "Lauren was a great student, I really wanted to get into teaching others to code, and not only did she listen and practice the skills I showed her, but she also gave me feedback to help my teaching abilities."

Hiring partners that were looking round said they were very impressed with how my week 6 team worked together, and really liked our approach of working to our weaknesses rather than strengths in a team, in order to improve ourselves. They also said the web page looked cool so I'm very happy with that.

I Am Equipped For Long Term Growth

I Manage My Own Wellbeing

What does it mean to be able to do this?

To manage your own wellbeing, in short, means to look after ourself. You're only human, and if the taxing nature of solving problems isn't enough to melt your mind after 20 minutes, then you still have the issue of staring at screens all day not being great for you. We've signed ourselves us for a really tricky job of basically being problem solvers, and all this brain power we're using can be drained very quickly. If you're anything like me, then the crying will take the last crumbs of energy you have left too hahaha. To look after yourself can take both physical and mental effort, in small steps, this could be taking a 5 minute break every hour, or making sure you ask for help after 20 minutes of being stuck, or even checkin in with yourself to see how you're feeling about a task. This is so important, as you're not going to do your best job if you're not your best self.

Evidence and explanation

I'd like to think I'm pretty good at looking after myself, however as this course went on, I found that actually I wasn't so great. I stupidly took the decision to try and lose weight when I started this course, something the coaches advised against on the first day too hahaha. I'd be eating significantly less and trying to fit in exercise when I can. This ended up in me not being able to focus at all for a good couple of weeks at makers. I then got to a point where I was so emotionally and physically drained, that I booked a meeting with Dana for a pep talk. Dana told me I was being way to hard on myself, and that I needed to praise my small wins rather than thinking I can't do something. After our chat, I decided that I really needed to eat more healthy food, and stop trying to go to the gym every evening, in order to have some peace. I also started attending meditation and talking to my cohort about my worries and returning the emotional support they gave me. After this, I found my quality of work improved massively, and I could focus a lot more. I go to Yoga almost every week, and I make sure that I have at least one day coding free on the weekends. Everyone needs a break, and that's okay.

Feedback

Kim: General Life - "You're great to talk to, you always give me a nice pep talk if I'm stressed and help ground me and plan out what I can do to solve my issues. I like how you're not afraid to ask me if you can vent or get my opinion on something too. If Yoda needs a day off I can always come to you."

I Can Learn Anything By Myself

What does it mean to be able to do this?

To be able to learn anything by yourself is a cool skill, to me it basically means you're equipped with the skills to be able to find / build your knowledge about any given subject matter. This isn't just googling what you don't know, or asking others for help and taking notes for you to use again, but also managing your learning in order to get it done effectively. An example of this is keeping a log of what you don't know, and how you'll go about doing what you can to know it.

Evidence and explanation

Around week 2 and 4, I had a complete meltdown about my progress. I felt like I couldn't do anything set without looking at the walk throughs and all my weekend challenges weren't going as well as my cohort's. I had a meeting with Ed in week 4 about what I could do to be better, going in with a little list of all the things I didn't know. And I asked him about effective ways to learn these topics. HE assured me that the fact I was making a list of things I don't know was a really good start, and I needed to focus more on how I'd obtain the knowledge I needed. The skill of being able to do it with no help would come with time, however this wasn't a vital requirement. After chatting with Ed, I realised that actually I was on a pretty good track with my learning, as I was able to identify what I did and didn't know, and what I needed to do to find out about it. There's nothing wrong with needing notes or assistance with applying these skills, as long as you have the ability to find the knowledge and assess your own abilities yourself. Following this revelation, I've kept a diary every week in my notebook of everything I don't know and what I need to do to learn it. The following week, I'd assess myself on how much I've improved with this skill. One particular thing I struggled with was doubles and mocking. After finding out that I didn't fully understand what they were, I was able to put in the effort to research them and ensure I knew their purpose. After that, I felt the next challenge was applying them, and I made sure to try and implement them when I could … after that, it's just practice making perfect.

Take a look at my Github, there's no way I would have been able to do all the things I've done if I didn't learn for myself. Also, I have a degree.. so I can learn for suresies.