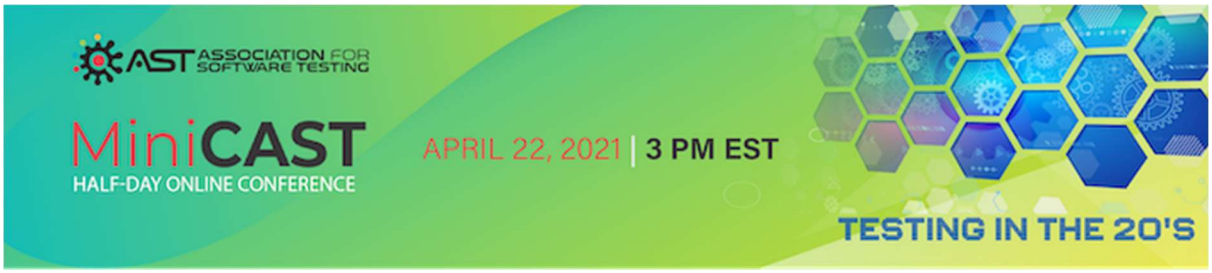

MiniCAST: Testing in the 20's

We're in a new decade now! A new wave of thinkers with something to say about testing in contemporary contexts is here.

MiniCAST is a half-day online conference produced by the Association for Software Testing that seeks to feel a little like CAST:

- Networking with smart and thoughtful testers – your peers
- Carefully curated non-commercial content
- Question and answer period after every talk

MiniCAST's program is drawn entirely from talks accepted for CAST 2020.

Schedule

MiniCAST starts at 3:00PM New York Time on Thursday, April 22.

3:00: Welcome!

3:15: Keynote: "Real Life is not an Edge Case" with Rachel Kibler

4:00: Networking Break

4:15: Track Talks: "Computational Thinking" with Sarah Aslanifar and "Testing Processes: One Size Does Not Fit All" with Bailey Hanna

5:00: Networking Break

5:15: Track Talks: "Developing Product Review Practices (That Work)" with Irja Straus and "Testing Machine Learning Models in Production" with Alex Eftimiades

6:00: Happy Hour Breakout

Real Life is not an Edge Case

In our testing, we often categorize flows and experiences into “happy paths” and “negative paths”, roughly following use cases, but finding weaknesses and risks in our unique ways. We sometimes think about personas, putting ourselves in the shoes of another, with details fleshed out about hobbies and coffee preferences. But we can also think about stress cases, which are where these personas meet the road. People have bad days, crises, and accidents all the time, and real life does not relegate them to the dreaded category of “edge cases”. Stress cases are when Facebook releases its “year in review”, intending to celebrate your year, but instead plastering your wall with pictures of your dead child. Stress cases are searching for an emergency room and receiving a Google Maps ad for urgent care. Stress cases are how to operate your phone when you’re carrying a squirming child or traveling in an elevator or tunnel. Stress cases are real life. And people are not edge cases.

This talk will present examples of stress cases and explore how we can expand our mindset and analysis to include them in our testing. We will discuss the overlap of accessibility with stress cases and how to enhance flows and UI to improve the experience for everyone. We will also talk about the limits of stress cases and the need for outside perspective.

Rachel Kibler

Rachel Kibler came to software testing as a third career, stumbling into it and finding that she enjoyed it. She now tests at 1-800 Contacts, working on a website team.

Rachel is the Treasurer and is on the Executive Board for the Association for Software Testing. In her free time, she makes music (usually voice and piano), knits, and plays board games. She also loves Harry Potter. Her website is http://racheljoi.com, and she tweets as @racheljoi.

Computational Thinking

We have a choice in designing our software development careers. We can follow the path of a technologist learning and exploiting one technology to solve the problems that particular technology was built to solve. Alternatively, we can be a computational thinker who can address a much broader set of problems, choosing or designing the solution needed for the problem at hand. Computational thinking is more than thinking like a programmer, and we'll see that not all programmers are thinking computationally.

Computational thinking is NOT thinking in programming language syntax. I think the worst thing we could do to get someone interested in programming is to introduce them to syntaxes like Java, C, or Fortran on day one. Why would they want to bother learning these unfamiliar, arcane details like pointers, classes, and compilation? Rather, let's start with a goal. What problem are they trying to solve? Can this problem be framed as a computational one that a computer can do much faster than we could?

Learning Outcomes:

- Explore the benefits of using computational models as a way to quickly and repeatedly create knowledge.
- Learn practical and free ways to practice computational thinking.
- Encourage others who are not programmers to think computationally when faced with problems that benefit from experimentation, fast feedback, automation, and forecasting.

Sarah Aslanifar

I was about 15 years old the first time I saw a computer, and I loved it! I joined a Math-focused track in high-school in Iran that'd prepare me for a Computer Science major in University. I came to the US at the age 18 and attended ISU, knowing very little English. I was admitted to their CS program shortly after I started my BS. I spent long days and nights at the computer lab until I was finally able to buy my own laptop before my graduation.

Over my seventeen year career as a software developer, I've served as a team lead in a Fortune 500 company, led teams globally as a consultant at ThoughtWorks, founded my own company, Tested Minds, lead a team at Morningstar to deliver CEO's highest priority project, and currently lead teams as a Principal Consultant at Tandem (madeintandem.com). My diverse roles have given me the chance to work with technical and business leaders with dozens of different styles. I love to share what I have learned with others!

Testing Processes, One Size Does Not Fit All

"So, what's your testing process like at *company name here*?". We've all been asked this question. It comes up all the time when you tell someone that you work in testing and who you work for. People want to know what your cookie cutter process is for testing at your company; either out of genuine curiosity, a search to find a fit for their skills or to find issues in it that they think you could solve by changing "just one thing". What happens when the answer is "we don't have one."? In the tech industry things are changing every single day, and every individual has their own skills and experiences that they bring to the table, so why would we have one specific process for our entire organization? In this talk, we will discuss what it looks like to have a constantly evolving test process within a company, how different teams can use the context they are working in to shape their own processes, and how to have multiple teams within a development org working with different processes and workflows effectively. We will discuss how we all stay on the same track, how we discuss our successes (and our failures), and how we use individual and team strengths to have an organic, and changing process that still keeps us all moving forward and working together.

Key Takeaways:
– Advantages and disadvantages of non-unified testing processes
– How to get buy in to use different processes across teams
– Tactics for determining best practices/processes for teams based on the context of the project and team members

Bailey Hanna

Bailey Hanna is a software test developer at D2L based out of Kitchener, Ontario. She has been an active member of the testing community for about 5 years, working to expand her knowledge and learn others in the software industry. Most recently she has joined the board of directors for the Kitchener Waterloo Software Quality Association to help give back to and grow her local testing community. Her primary areas of passion and experience are context driven exploratory testing, test strategy and the development of automated tests. Through her path into and through testing she's learned a lot about good practices, and of course bad ones, and looks forward to opportunities to share these learnings with others.

Developing Product Review Practices (That Work)

Have you ever wondered how you, as testers, could give a helping hand even *before* a single line of code is written?

In this talk, I will *eat my own dog food* and use my mistakes as examples to demonstrate why including (and *accepting*) a tester's critical feedback helps to assess risks in a timely manner. I see the product with different eyes when I am in different roles. Whether I'm a product manager or tester, I have had different biases. This talk is also a story about the practices I use today when performing requirements and design reviews (or simply, product reviews). Those practices enable me to ask the right questions before development efforts have started and when it's not (so) expensive to address them.

I will explain some of the heuristics I use to detect if there is a possibility of *reinventing the wheel*, having a "not great, not terrible" user experience in particular product areas, or is it too easy for a user to make an undoable mistake. The heuristics can also help in finding any vague or missing requirements, inconsistent rules and behaviors, and project and product risks. I like to use those heuristics as "conversation starters" to perform a risk-based product review, and this talk can also help you learn how to apply them to your contexts.

After listening to this story, I would like you to take home a few new tools in your toolbox that strengthen the testing skills which can be used when performing product reviews. Hopefully, my experience will inspire you to rethink the ways you can help and bring more value to your products and people.

Irja Straus

After her adventure in product management, Irja has gone back to her first love – Software Testing. She's currently a Senior Quality Control Analyst and Team Leader in Infobip.

She likes to learn the human and business side of software, experiment with products to investigate risks, and advocate bugs that matter. Software testing is what she loves to do, and she never designed the same test strategy twice in her 11-year career.

Testing Machine Learning Modules in Production

Standard software testing works just fine when you know your outputs. What happens when success means “at least 95% accuracy at least 90% of the time”? Worse still, what happens when success means “This group liked the analysis it gave, so it’s fine for stuff that looks like what came in yesterday”? In this talk, we will share our experiences building FINRA’s first frameworks, guidance, and case studies on monitoring machine learning models in production.

Upon building context around how machine learning models are evaluated, we share our experience developing and applying some of our most promising approaches to monitoring model predictions as well as data quality.

Alex Eftimiades

Alex is a data scientist at FINRA, working jointly between the quality assurance department and another group within Technology that researches anomalies that may be indicative of insider trading and fraudulent activity. He applies machine learning and statistics to address these areas. Alex originally studied physics and is passionate about applying math to solve real world problems. He previously worked as a data engineer and software engineer.