Agile Testing Essentials – Resources for More Learning

General resources:

- "Quick Tools for Agile Testing" available in 10 languages: http://agiletester.ca/quick-tips-for-agile-testing/
- Book website for *Agile Testing: A Practical Guide for Testers and Agile Teams* and *More Agile Testing: Learning Journeys for the Whole Team*, including a downloadable Chapter 8, Using Models to Help Plan from *More Agile Testing*, http://agiletester.ca/
- Lisa Crispin's blog: lisacrispin.com/
- Janet Gregory's blog: janetgregory.ca/blog/

Acronyms defined:

ATDD - Acceptance-test-driven development. Teams doing ATDD specify executable acceptance tests using examples elicited from conversations with customers. Each example is discussed and, if possible, specified in an executable format. Programmers use the tests as guidance to develop the feature. Once code is written to make a test pass, it may be kept as an automated regression test, documenting feature behavior.

BDD - Behavior-driven development. Examples of desired behavior from conversations with stakeholders are turned into executable tests in a given-when-then format. Programmers automate the tests, then write the code to make each test pass. BDD can be used at the unit level for test-driven design and can be used for business-facing tests that encompass multiple components and layers of the application and verify business-facing quality.

CI – Continuous Integration. It is a process of integrating code from multiple developers or multiple teams of developers into a shared repository several times a day.

CD – Continuous Delivery. As defined by Martin Fowler, it is a software development discipline where you build software in such a way that the software can be released to production at any time. See Martin Fowler's website at https://www.martinfowler.com/bliki/ContinuousDelivery.html.

DSL – Domain Specific Language. It's used to describe examples of desired and undesired system behaviour in language relevant to the company's business, that business stakeholders understand as well as technical team members.

ET – Exploratory Testing. Exploratory testing combines test design with test execution and focuses on learning about the application under test.

MVP - Minimum viable product. The term minimum viable product was coined by Frank Robinson and popularized by Eric Ries for software products (Wikipedia, 2014). A minimum viable product has just enough core features to allow the product to be deployed to a small subset of potential customers. These users form part of the "build-measure-learn" or "validated learning" feedback loop, guiding further development and decision making. It is also used to represent the least amount of change that can be delivered to the customer that adds business value.

Q1, Q2, Q3, Q4 – the four Agile Testing Quadrants: technology-facing tests that guide development, business-facing tests that guide development, business-facing tests that critique the product, and technology-facing tests that critique the product, respectively. See http://agiletester.rumspeed.net/wp-content/uploads/sites/26/2014/09/Gregory_Chapter_8_Final.pdf for more on the Agile Testing Quadrants.

QA - Question Asker. A software professional who engages in testing activities.

SBE - Specification by example. Specification by example uses process patterns that help software delivery teams collaborate with customer teams to define the scope of work to achieve business goals. SBE illustrates specifications with concrete examples, refines the specification from key examples, and then automates them to validate the specifications frequently during development. The validated executable specifications are kept as living documentation (Gojko Adzic, 2011).

TDD - Test-driven development. In test-driven development, the programmer writes and automates a small unit test, which initially fails, before writing the minimum amount of code that will make the test pass. The code is refactored as needed to meet acceptable standards. The production code is made to work one test at a time. TDD, also known as test-driven design, is more of a code design practice than a testing activity and helps build robust, easily maintainable code.

Frameworks mentioned:

- Cucumber: https://cucumber.io/
- Robot Framework: http://robotframework.org/
- FitNesse: http://fitnesse.org
- Fit: http://fit.c2.com/
- Selenium WebDriver: http://www.seleniumhq.org/projects/webdriver/
- Watir: https://github.com/watir/watir

Tools mentioned:

- Bug Magnet chrome extension: http://gojko.github.io/bugmagnet/
- Impact mapping: http://impactmapping.org
- MindMup: https://www.mindmup.com/
- Test Heuristics Cheat Sheet: http://testobsessed.com/wp-content/uploads/2011/04/testheuristicscheatsheetv1.pdf

Books, articles, and blog posts mentioned:

- Adzic, Gojko, *Specification by Example,* Manning 2011
- Cohn, Mike, *The Forgotten Layer of the Test Automation Pyramid*, https://www.mountaingoatsoftware.com/blog/the-forgotten-layer-of-the-test-automation-pyramid 2009
- Emery, Dale H., *Writing Maintainable Automated Acceptance Tests*, 2009, http://dhemery.com/pdf/writing_maintainable_automated_acceptance_tests.pdf
- Gottesdiener, Ellen and Mary Gorman, *Discover to Deliver: Agile Product Planning and Analysis*, 2012
- Gregory, Janet, *View into test planning at the release leve*l, 2012, http://janetgregory.ca/view-into-test-planning-at-the-release-level-2/

- Gregory, Janet, *Jigsaw Puzzles & Small Chunks*, 2011, http://janetgregory.ca/jigsaw-puzzles-small-chunks/
- Hendrickson, Elisabeth, *Explore IT! Reduce Risk and Increase Confidence with Exploratory Testing*, Pragmatic Programmer, 2013
- Knight, Adam P., *T-shaped Tester, Square Shaped Team*, Guest blog on The Social Tester; http://thesocialtester.co.uk/t-shaped-tester-square-shaped-team/
- Manns, Mary Lynn, and Rising, Linda*, More Fearless Change: Strategies for Making Your Ideas Happen,* Addison-Wesley, 2015
- Marick, Brian, Original blog post about the agile testing matrix (quadrants), 2003, http://exampler.com/old-blog/2003/08/21.1.html#agile-testing-project-1
- North, Dan, *Introducing BDD*, 2006, https://dannorth.net/introducing-bdd/
- Patton, Jeff, *Pragmatic Personas: Putting the User back in User Stories*, 2009, https://www.infoq.com/presentations/pragmatic-personas
- Patton, Jeff, *Story Mapping Slides*, 2016, http://jpattonassociates.com/storymappingslides/
- Patton, Jeff, *User Story Mapping: Discover the Whole Story, Build the Right Product,* O'Reilly, 2014
- Pyhäjärvi, Maaret, *Amplified Learning with Mob Testing*, Stickyminds, 2017, https://www.stickyminds.com/article/amplified-learning-mob-testing
- Pyhäjärvi, Maaret, *A Mob Testing Experience*, 2017, http://visible-quality.blogspot.de/2017/03/a-mob-testing-experience.html
- Wynne, Matt, *Introducing Example Mapping*, https://cucumber.io/blog/2015/12/08/example-mapping-introduction
- Vydra, David, *Using just enough automation to dramatically improve the testing process*, 2011, http://www.testdriven.com/2011/05/using-just-enough-automation-to-dramatically-improve-the-testing-process/

More reading:

- Adzic, Gojko, *How to get the most out of Given-When-Then*, 2015, https://gojko.net/2015/02/25/how-to-get-the-most-out-of-given-when-then/
- Barile, Jason, *Don't Assume If Something Worked Once, It'll Work Again*, 2011, https://blogs.msdn.microsoft.com/jasonba/2011/07/12/dont-assume-if-something-worked-once-itll-work-again/
- Crispin, Lisa and Nanda Lankalapalli, *Decrease Your Debt with Technical Debt Sprints*, StickyMinds 2011 http://www.stickyminds.com/sitewide.asp?Function=WEEKLYCOLUMN&ObjectId=16827&ObjectType=ARTCOL&btntopic=artcol
- Knight, Adam P., *The Thread of an Idea: Adopting a Thread-Based Approach to Exploratory Testing*, 2011, http://www.a-sisyphean-task.com/2011/11/thread-based-approach-to-exploratory.html
- Pyhäjärvi, Maaret and Llewellyn Falco, *Mob Programming Guidebook,* 2016, https://leanpub.com/mobprogrammingguidebook