About these slides

This presentation contains modified content from the presentation “Adventures in Modern Testing” ([http://bit.ly/mt-pnsqc](http://bit.ly/mt-pnsqc)) by Alan Page licensed under [CC BY 4.0](http://creativecommons.org/licenses/by/4.0)
Alan Page

- Spent 22 years at Microsoft
- Works for Unity Technologies since 2017
- Alan’s Blog: https://angryweasel.com/blog
- AB Testing Podcast: https://www.angryweasel.com/ABTesting/
My Background

Studied CS at University of Tartu like you

Software development industry experience since 2006

Giving test automation seminar “Hands-on Software Testing” course at UT
Hiking

Always looking to acquire cool new gear.
Boardgames

Currently playing Gloomhaven (pictured)
This is not a future of testing talk
This is a talk about embracing change

It is not the strongest or the most intelligent who will survive but those who can best manage change. --Charles Darwin
My Evolution as a Software Tester
Nokia (Plenware/Cybercom) 2006-2008

- Separate outsourced near-shore test team
- Only regression testing
- Automating a big chunk of the regression scope
Playtech (2009-2013)

- QA physically separated from Dev
- Extensive manual black-box testing
- Automating regression testing of Flash games
Skype (2013-2014)

Pair programming + TDD

Integration tests written before writing business logic

Whole team regression testing

I realize that developers can become very good at testing

We're a team!
RSI (2014 - Present)

- Most test automation by testers
- Testers sometimes the bottleneck
- Manual testing done only by team’s QA specialist

We’re a team too!

QA
DEV
DEV
DEV
DEV
DEV
DEV
Who is this mysterious team, who lives on foreign land and has been sending us all of these bugs!?

We're a team!

QA

We're a team too!

Dev
The inevitable evolution of the Test Role...
The inevitable evolution of the Test Role...

We're a team!
Crossing the Chasm
"Transition"

Separate Teams

More SDETs/Automators

More "Manual" Testing

Modem Testing Agile Testing Traditional Testing

Low-Skilled Trad Testing
"Transition"

Separate Teams

More SDETs/"Automators"
More "Manual" Testing

Modem Testing
Agile Testing

Traditional Testing

Low-Skilled Trad Testing

RSI
SKYPE
PLAYTECH
NOKIA
So What Exactly Is Modern Testing?
Modern Testing is the Antidote for “Traditional” Testing Methods
“Traditional Testing”

- Separate (and often siloed) Test team
- Concerned with consistency & predictability
“Agile Testing”

- Test Specialist on Feature Team
- Works primarily on testing tasks needed for feature / product quality
“Modern Testing”

- Test Specialist on Feature Team
- Data-driven and customer obsessed
- Passionate about efficiency
“Traditional” Testing May Harm Business Goals By:

- Creating unnecessary delays and costs
- Being focused on code and specification correctness instead of quality
- Enabling a ‘safety net’ culture and failing to deliver on the promise of moving quality upstream
- Treating test activities as an innate specialization that cannot be taught
- Favoring isolationism over community
Introducing the Modern Tester
Focuses on speeding up the Team

Equal member of the Feature Team

Test/Quality Coach

Specializing Generalist

Leverages customer data extensively

Always Learning

Does some testing
Modern Testing Principles #1

Our priority is improving the business
Modern Testing Principles #1

- *Testing contributes positively to the business by reducing costs and increasing revenue*
- *Priority is delivering quality and customer satisfaction (not code / functional correctness)*
Modern Testing Principles #2

We accelerate the team, and use models like Lean Thinking and the Theory of Constraints to help identify, prioritize and mitigate bottlenecks from the system
Principle #2 Accelerating the Team

- Eliminate inefficiency
- Find and mitigate bottlenecks
- Remove delays
- Know / apply lean principles
Principle #3

We are a force for continuous improvement, helping the team adapt and optimize in order to succeed, rather than providing a safety net to catch failures.
Principle #3 - Continuous Improvement

- *Help the team learn from mistakes and adapt*
- *Testers are NOT the last line of defense*
- *Being a safety net is a bottleneck (see principle number 2)*
Principle #4

We care deeply about the quality culture of our team, and we coach, lead, and nurture the team towards a more mature quality culture
Principle #4 - Quality Culture

- Quality culture is a shared mindset
- Requires leadership, communication, tools, learning, and desire to improve
- It’s a people problem
Quality Culture - Potential Attributes

- Quality Ownership
- Breadth of Testing
- Technical Debt and Maintenance
- Code Quality and Tools
- Development Approach
- Data usage
- Leadership
Principle #5

We believe that the customer is the only one capable to judge and evaluate the quality of our product
Principle #5

- Who is your customer? HINT: you are not the customer
- Are you building the right product?
- Focus on the business (principle number 1)
- Collect data
Principle #6

We use data extensively to deeply understand customer usage and then close the gaps between product hypotheses and business impact.
Principle #6 - Data

- Understand the real customer experience
- Shipping is a hypothesis
- Learn and adapt
Principle #7

We expand testing abilities and knowhow across the team; understanding that this may reduce (or eliminate) the need for a dedicated testing specialist.
Principle #7 - Expand testing knowhow across the team

- **Software testing isn’t a dark art**
- **Focus on the first half of this principle. If the second half is meant to be, it will happen naturally**
- **Generalizing is good**
The Modern Tester is a Specializing Generalist
Accelerate the Achievement of Shippable Quality
What next? Is test dead? Meet the modern tester

The role of the sapient tester is changing, but automation will not take over jobs of testers
Testing vs. checking

Checking Is Confirmation

Testing is Exploration and Learning

Source: http://www.developsense.com/blog/2009/08/testing-vs-checking/
When should we stop automating?

Are there any bugs we could miss even if we have a very high coverage from automated tests?

What kind of bugs we might miss with test automation?

When will they learn? It’s not about getting more traps, I just don’t like cheese.

It is not enough to be busy, the question is what are we busy about?
- Henry David Thoreau

Visiting the Ancient Pyramid* of Test Automation

*Note, like the actual pyramids in Egypt, the levels of automation pyramid is a bit out of fashion already, but we still talk about it from time to time.

Image: https://martinfowler.com/bliki/TestPyramid.html
Continuous integration (CI) and continuous deployment (CD)

<table>
<thead>
<tr>
<th>ID</th>
<th>DEPLOY BY</th>
<th>COMPONENT</th>
<th>BRANCH/PULL REQUEST #</th>
<th>REVISION</th>
<th>TEST DEPLOY</th>
<th>TESTS</th>
<th>LIVES DEPLOY</th>
<th>LIVE TESTS</th>
<th>INFO</th>
<th>QUEUED SINCE</th>
<th>TOTAL TIME</th>
<th>UPDATED</th>
</tr>
</thead>
<tbody>
<tr>
<td>9957</td>
<td>dorishalimagi</td>
<td>webapp</td>
<td>RED-260oct-quickfix pr #4609</td>
<td>success 6m 16s</td>
<td>success 57s</td>
<td>no tests</td>
<td>started</td>
<td>pending</td>
<td>pending</td>
<td>Started TEST jenkins tests</td>
<td>12h ago</td>
<td>5m ago</td>
</tr>
<tr>
<td>9956</td>
<td>devuo</td>
<td>superdata-sapi</td>
<td>NAVY-62-person CRUD pr #103</td>
<td>success 4m 55s</td>
<td>success 9s</td>
<td>success 1m 45s</td>
<td>success 20s</td>
<td>success 52s</td>
<td>no tests</td>
<td>success after 6m</td>
<td>6m 26s</td>
<td>1h ago</td>
</tr>
<tr>
<td>9955</td>
<td>magnuspipe</td>
<td>partners-portal</td>
<td>NM-415 pr #72</td>
<td>success 3m 4s</td>
<td>success 15s</td>
<td>no tests</td>
<td>no tests</td>
<td>success 1m 31s</td>
<td>success 18s</td>
<td>success after 5m</td>
<td>5m 9s</td>
<td>3h ago</td>
</tr>
<tr>
<td>9954</td>
<td>jaanuspoldre</td>
<td>Pipedrive</td>
<td>remove-debug-msg pr #6905</td>
<td>success 1m 28s</td>
<td>n/a</td>
<td>n/a</td>
<td>no tests</td>
<td>no tests</td>
<td>success 10m 48s</td>
<td>success after 17m</td>
<td>5m 26s</td>
<td>4h ago</td>
</tr>
<tr>
<td>9953</td>
<td>vitality-komenda</td>
<td>backoffice-ui</td>
<td>DARK-429 pr #82</td>
<td>success 6m 9s</td>
<td>success 14s</td>
<td>no tests</td>
<td>no tests</td>
<td>success 23s</td>
<td>success 10s</td>
<td>success after 6m</td>
<td>6m 57s</td>
<td>3h ago</td>
</tr>
<tr>
<td>9952</td>
<td>hhabakuk</td>
<td>contacts-timeles-service</td>
<td>AQUA-fx Logging pr #45</td>
<td>success 3m 38s</td>
<td>success 9s</td>
<td>no tests</td>
<td>no tests</td>
<td>success 1m 14s</td>
<td>success 18s</td>
<td>success after 5m</td>
<td>5m 26s</td>
<td>4h ago</td>
</tr>
<tr>
<td>9951</td>
<td>koita</td>
<td>company-statistics</td>
<td>RUBY-102 pr #263</td>
<td>success 4m 4s</td>
<td>success 9s</td>
<td>no tests</td>
<td>no tests</td>
<td>success 16m 1s</td>
<td>success 32s</td>
<td>success after 23m</td>
<td>23m 55s</td>
<td>3h ago</td>
</tr>
</tbody>
</table>
Testing in Production (TiP)

I see you test your code in production.

I too like to live dangerously.
Testing in Production (TiP)

- Not the testing testers would do in production environment
- Testing done by the end-users
- Canary deployments:
  - Usually a new version is released to a small percentage of users and then its quality is closely monitored from several different metrics: logs, business intelligence tools etc.
- Needs good infrastructure support for easy rollbacks and alerting for times when things go bad
Who is a modern tester? Predictions by Alan Page

- Independent test teams are diminishing in favor of the test specialist - think cross-functional teams composed of specialists
- They will write less UI automation
- Testers writing application automation will become a rarity
- Huge amounts of testing will be done via monitoring real customer usage - remember TiP

Source: http://angryweasel.com/blog/predictions-and-other-stupid-things/
The greatest tools

“The future is already here — it's just not very evenly distributed.”

- William Gibson
Thank you