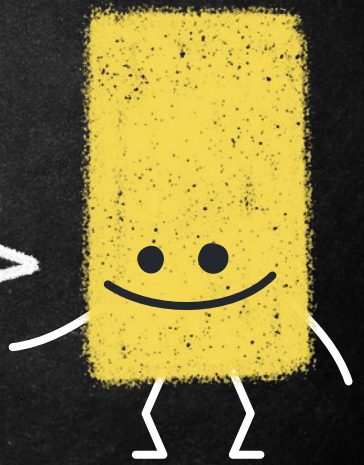
AGILE TESTING DAYS USA

Software investigation & exploration – Supercharging your value

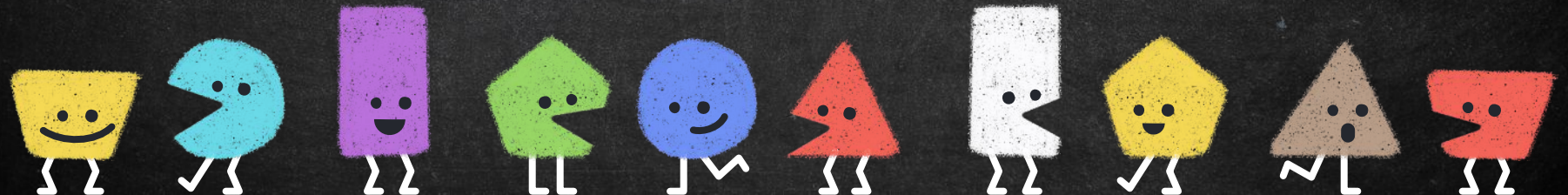


Hello!

I am Huib Schoots

I am here because I love to do workshops.

You can find me on twitter at **@huibschoots**

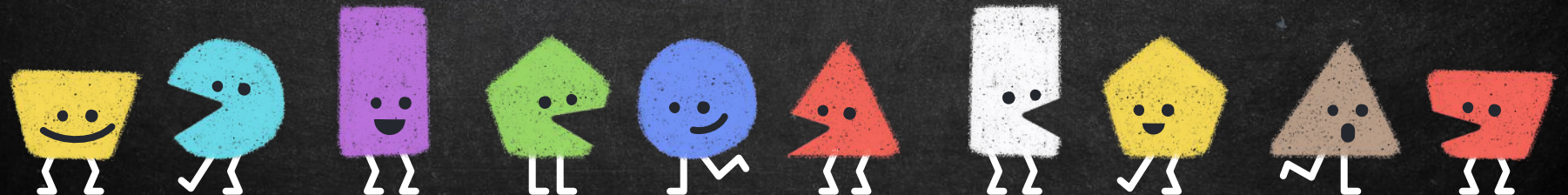


Hello!

I am Alex Schladebeck

I am here because I love Huib.

You can find me on twitter at **@alex_schl**



What do you what to learn?

Goals for today

- Explain what we think (exploratory) testing is
- Learn by doing lots of fun exercises where we test real software.
- Learn essential skills like exploration, note taking, use of heuristics and chartering
- Learn to become a better tester, not by teaching you how to test, but by letting you experience what you do!
- You will learn to recognise what you do while you are testing. You can use that as a learning tool to get better at testing.

First quick exercise...

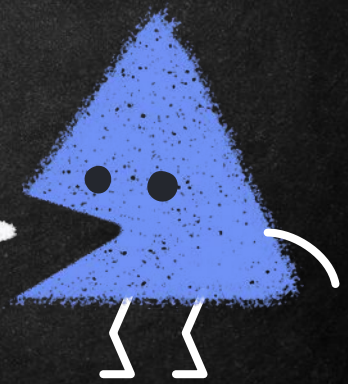
What is ET anyway?

→ What is exploratory testing?

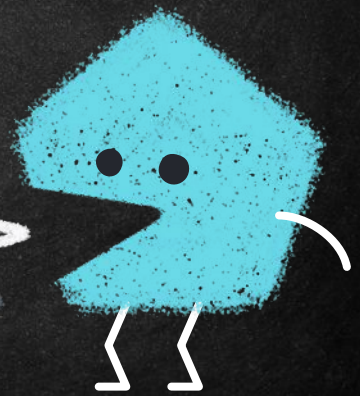
→ What is the goal of ET?

→ Who does it?

Write down your answers in 3 minutes

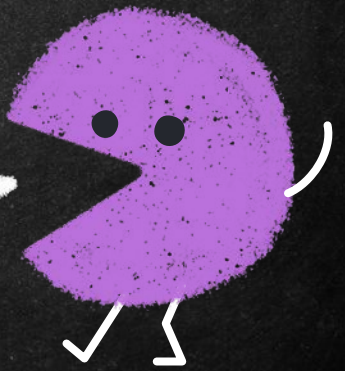


What makes ET hard?

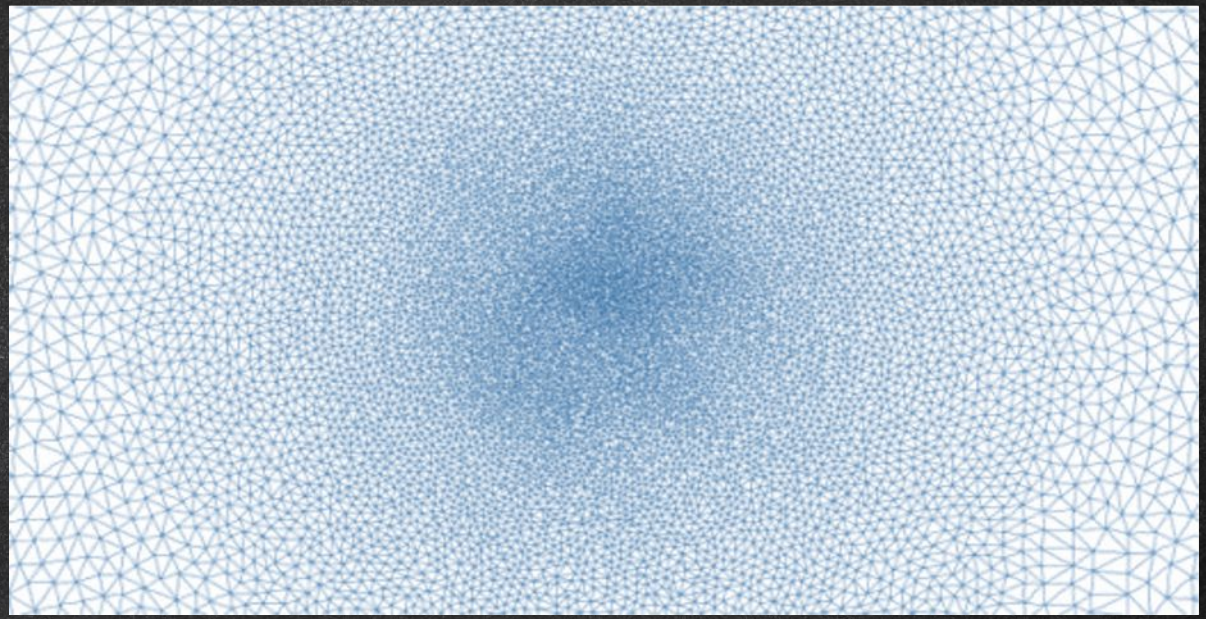


Let's talk about software development

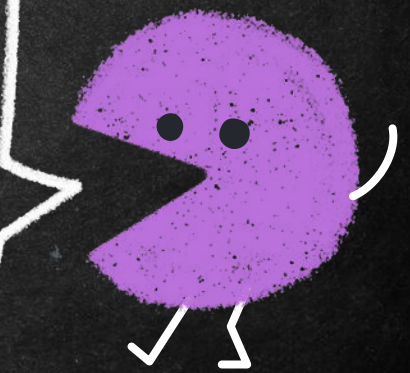
Software development =
RESEARCH & Development



Software is complex



People are complex



@huibschoots @alex_schl

Dealing with this VUCA world

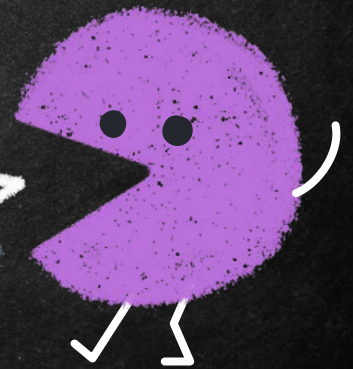
- → Volatile — Product vision
- → Uncertain — Try to understand the context
- → Complex — Learning, insight and providing clarity
- → Ambiguous — Agility to incorporate adaptability

RISK!

During software development we have to deal with unknown unknowns

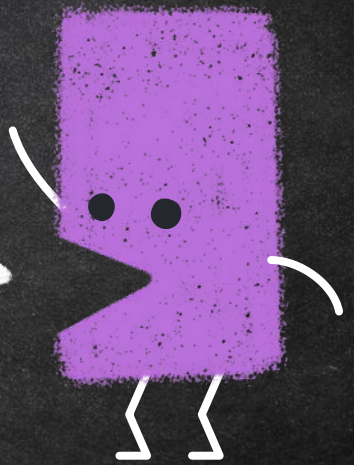
- → Quality is perception
- → Customers and product owners don't know or can't imagine what they want
- → Development team can't image what customers will actually do
- → Research: building new insights & evolutionary design
- → Dealing with complexity, confusion, change, new insights and half answers

We have to learn and to deal with risks!



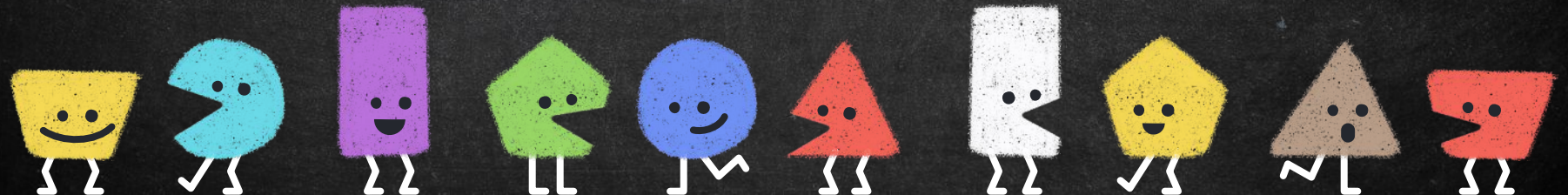
“

The biggest risk in software development is the risk of overlooking or missing an important risk!



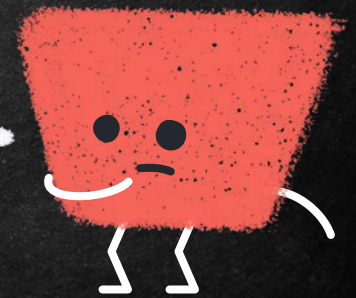
STORY BY ALEX

HEY ALEX, HOW WOULD YOU EXPLAIN EXPLORATORY TESTING?



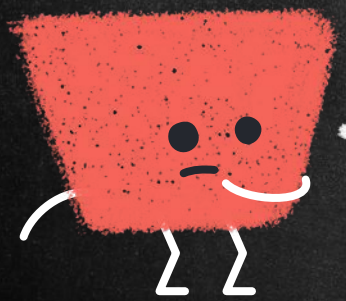
1. Manual testing takes ages

- Designing and maintaining test cases takes a great deal of time
- The tests also need to be executed (often manually)
- The people who execute the tests often aren't the people who wrote them
  - So there are differences and questions
  - Or we attempt to add so much detail to the test cases — which results in even more work



2. Scripted tests don't find new problems

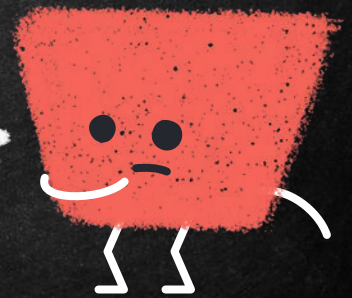
- Boris Beizer: "Every method you use to prevent or find bugs leaves a residue of subtler bugs against which those methods are ineffectual."
- Test cases become less and less effective over time.
- They need to be updated, added to, deleted, changed
- Even then, we cannot hope to write and execute and maintain *all known* test cases for software



3. Scripted tests don't help us find unknown risks

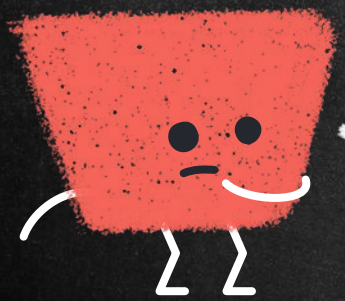
- We can only write test cases for risks that we have identified
- And the test cases for these identified risks become less effective in finding problems (previous slide)
- And it's usually the things we didn't think about that cause problems
- Think of a net: the individual strings are test cases. But there will always be holes

Elisabeth Hendrickson, Explore It!

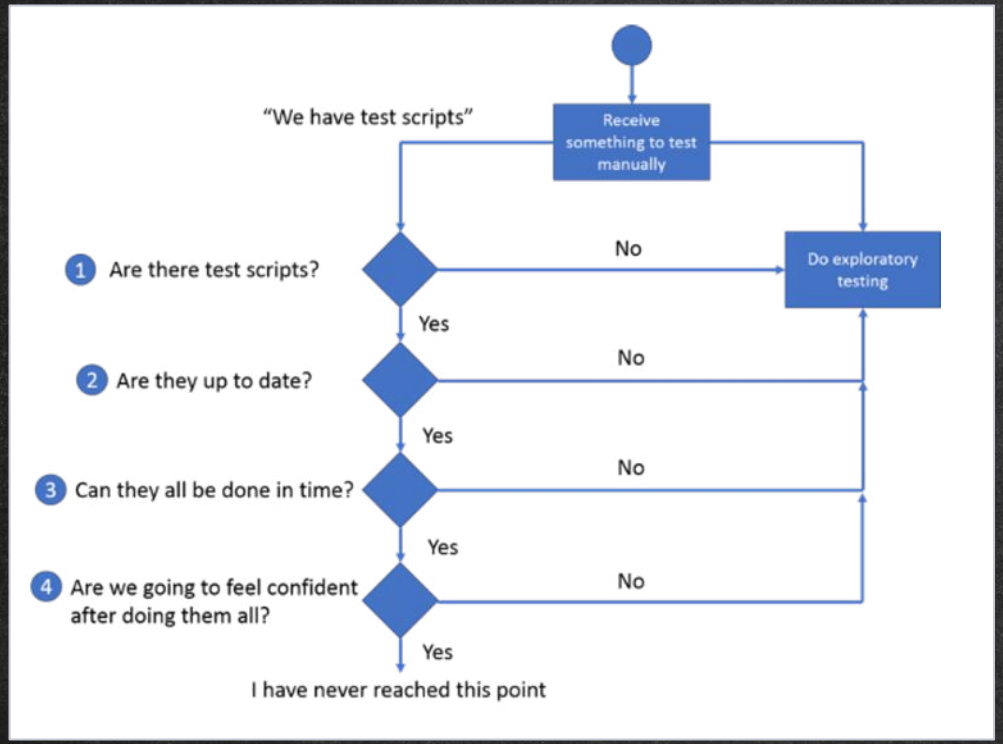


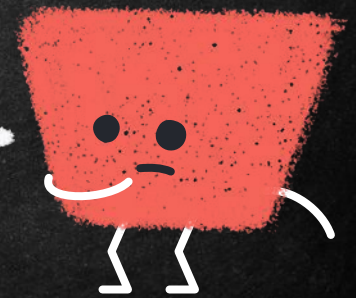
4. Observational blindness

- Inattentional blindness is the psychological phenomenon that causes you to miss things that are right in front of your eyes.



5. ET will always be necessary



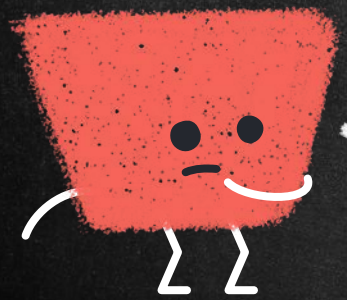


6. And something about pelicans...

[EXPLORATORY TESTING] IS A THING THAT ALWAYS HAS TO HAPPEN BECAUSE WE CAN AUTOMATE AS MUCH AS WE LIKE BUT WE CAN ONLY AUTOMATE JUST RISKS WE HAVE SEEN AND THERE ARE ALWAYS GOING TO BE RISKS THAT WE HAVEN'T SEEN.

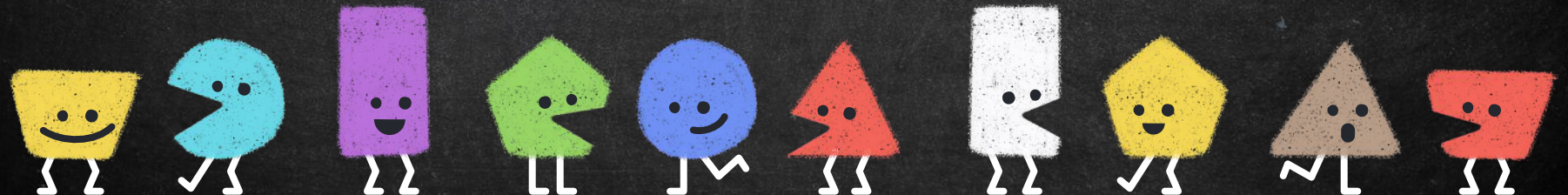
MY CURRENT FAVORITE EXAMPLE OF THIS IS: I WENT RUNNING IN A PARK VERY EARLY IN THE MORNING. OBVIOUSLY, I HAVE VARIOUS RISKS I CAN THINK ABOUT: I MIGHT INJURE MYSELF, I MIGHT GET LOST, I MIGHT GET ATTACKED. THOSE ARE THINGS I THOUGHT ABOUT. AS IT TURNED OUT, IN THE PARK THEY HAVE A PELICAN. I APPROACHED THIS PELICAN TO GET A PHOTO OF IT.

THE PELICAN MADE A THREATENING MOVE TOWARDS ME AND I RAN AWAY, TERRIFIED. ON THE LIST OF RISKS THAT I HAD IN THAT PARK ON THAT MORNING, BEING ATTACKED BY WILDLIFE WAS NOT ONE OF THEM. AND THAT IS ONE OF MY BEST EXPLANATIONS OF WHY WE NEED EXPLORATORY TESTING.
– ALEX SCHLADEBECK

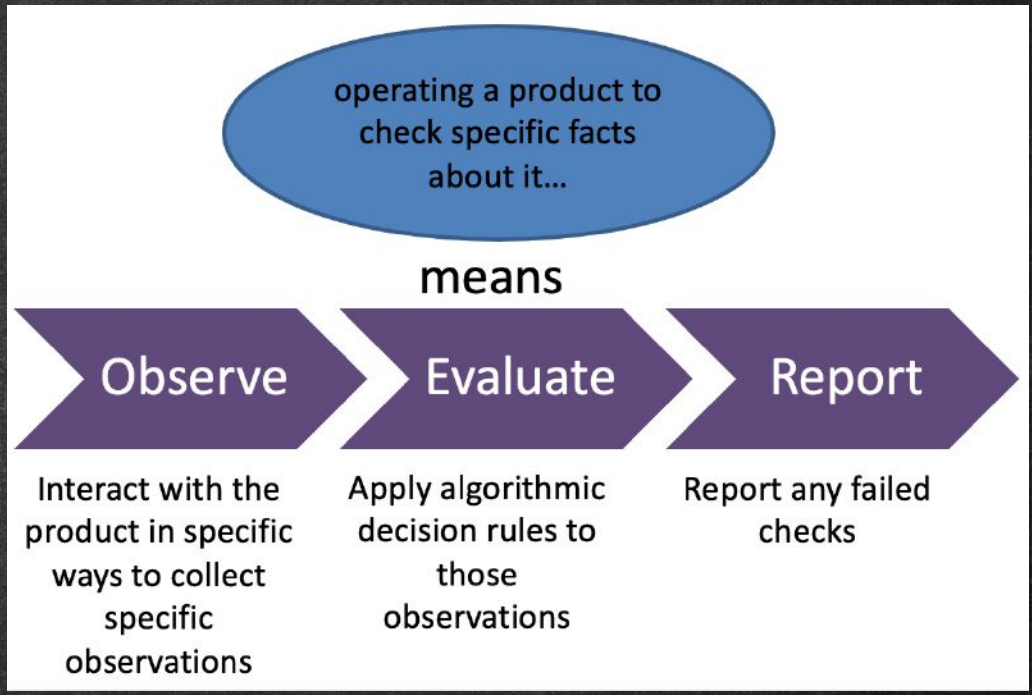


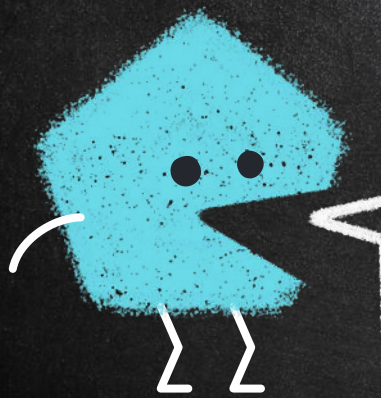
STORY BY HUIB

SO HUIB, DO YOU HAVE ANYTHING TO ADD TO THAT STORY?



Call this "Checking" not Testing





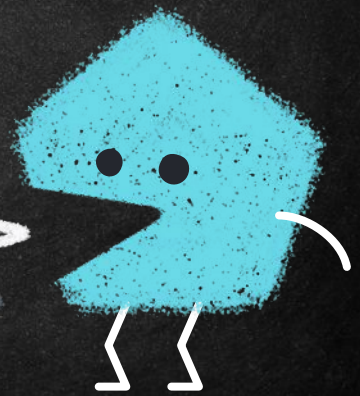
A Check Has Three Elements

1. An *observation* linked to…
2. A *decision rule* such that…
3. both observation and decision rule can be applied algorithmically.

A ***check*** can be performed

by a machine
that *can't* think
(but that is quick and precise)

by a human who has been
instructed *not* to think
(and who is slow and variable)



Testing is...

Acquiring the competence, motivation, and credibility for...

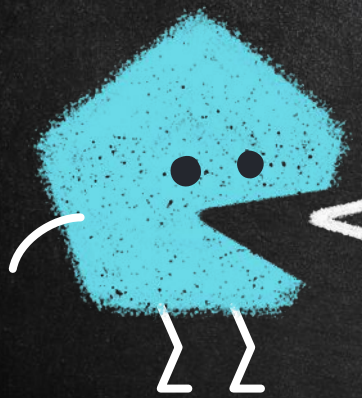
creating the conditions necessary for...

evaluating a product by learning about it through experimentation, which includes to some degree: questioning, study, modeling, observation and inference, including...

operating a product to check specific facts about it...

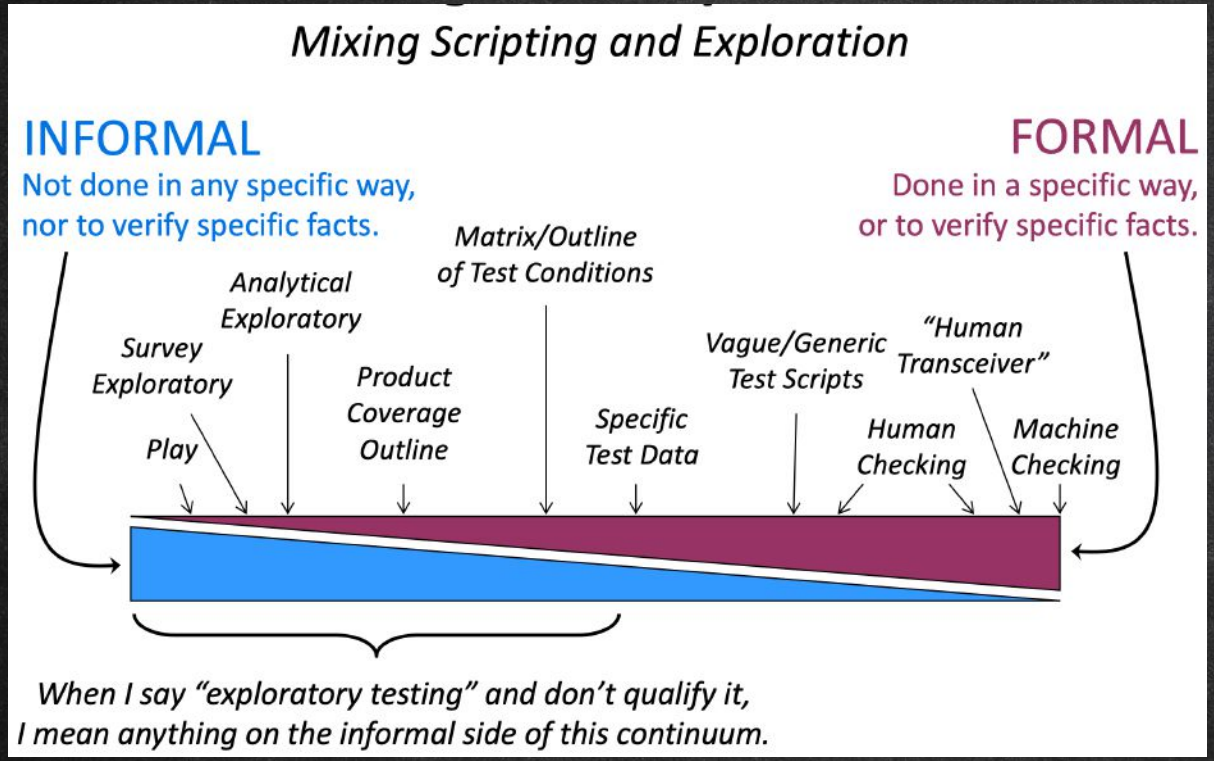
...so that you help your clients to make informed decisions about risk.

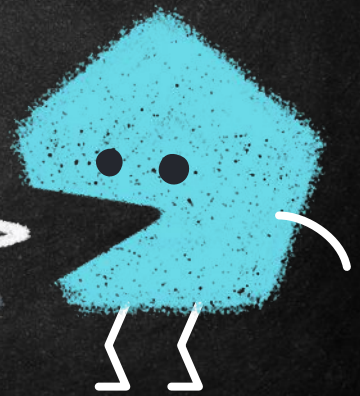
And perhaps help make the product better, too



@huibschoots @alex_schl

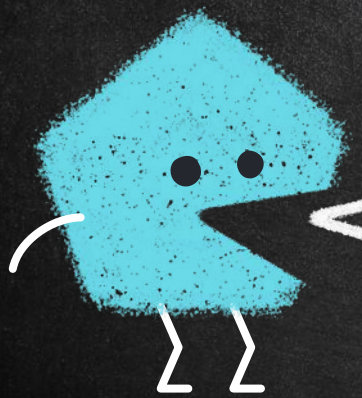
The Testing Formality Continuum

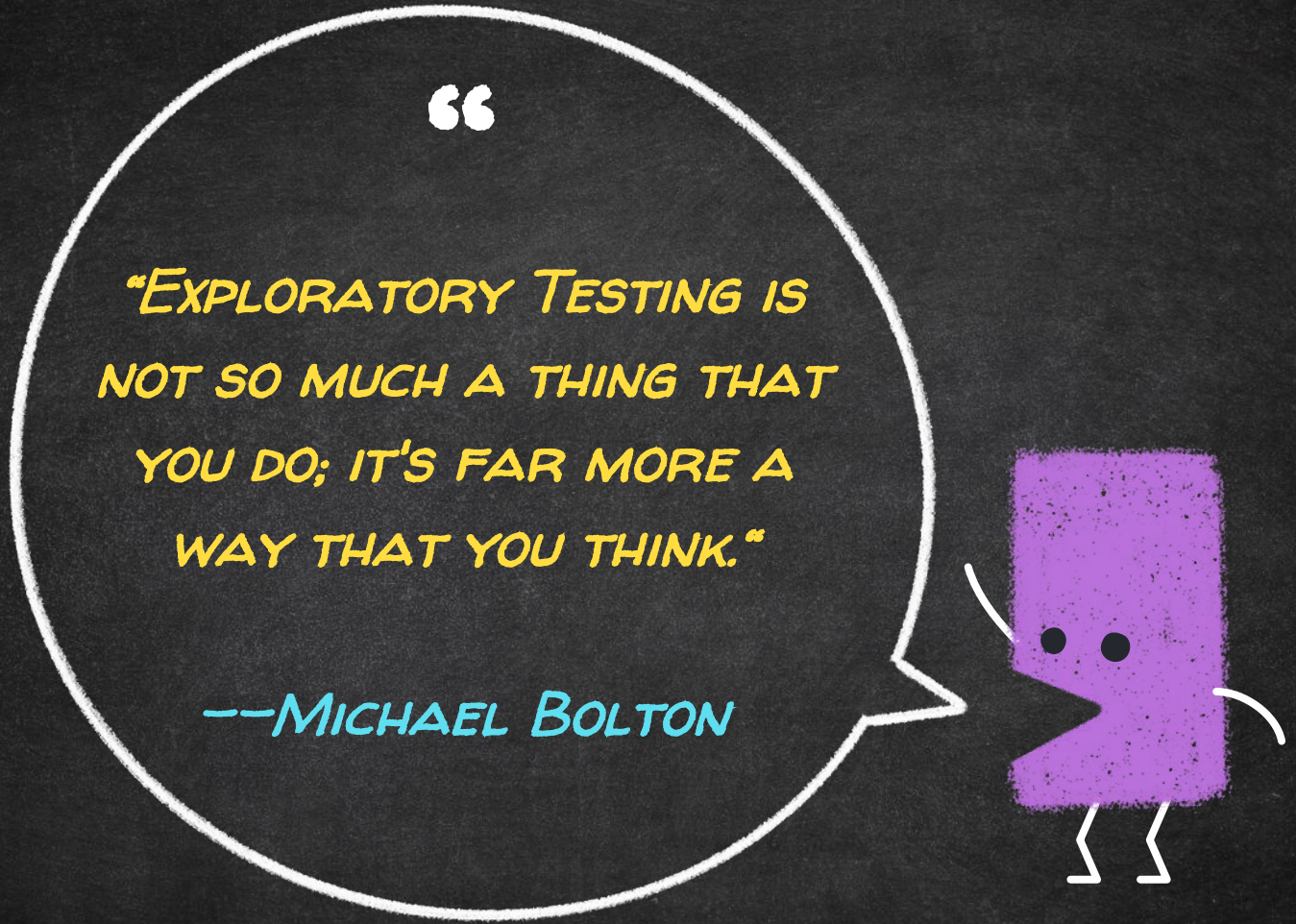




Myths & misconceptions

- ET is unstructured
- ET is undocumented
- ET doesn't provide accountability
- ET cannot be measured
- ET is for experienced people only
- Scripted testing is easier to do than ET
- No review in advance
- Hard to repeat
- Difficult to report test coverage



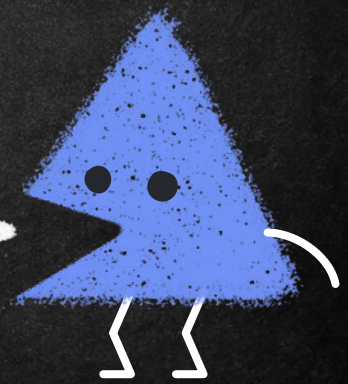

“
“Exploratory Testing is not so much a thing that you do; it's far more a way that you think.“
--Michael Bolton

Let's do a testing exercise...

→ Use this application:
http://demowebshop.tricentis.com/

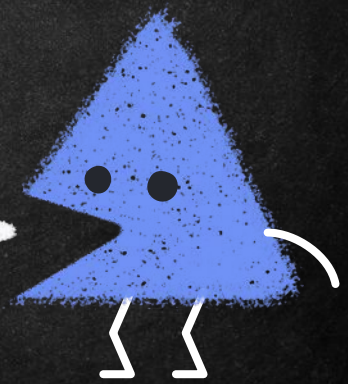
Test it for 20 minutes

We will debrief together

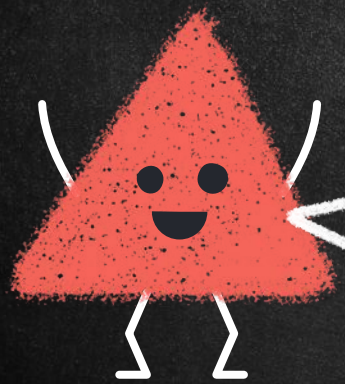


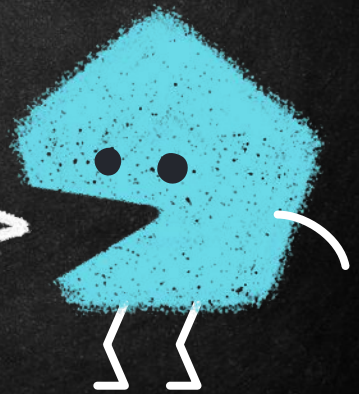
Credit Cards Details

Type	Number	Exp date	Check ID
Visa	4444 3333 2222 1111	03/2030	737
MasterCard	5555 4444 3333 1111	03/2030	7373
Amex	3700 0000 0000 002	03/2030	7373
Diners	3607 0500 0010 20	03/2030	737
Discover	6445 6445 6445 6445	03/2030	737
Visa Electron	4001 0200 0000 0009	03/2030	737
Maestro	6771 7980 2100 0008	03/2030	737

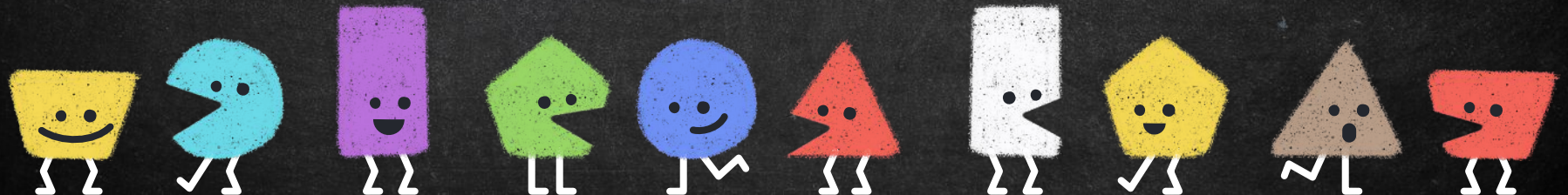


Debrief





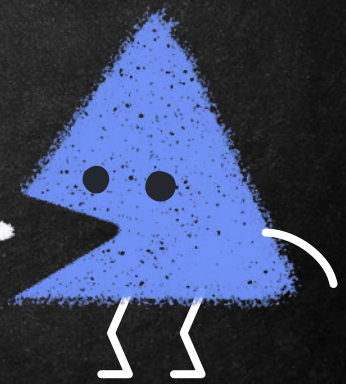
Structuring your testing



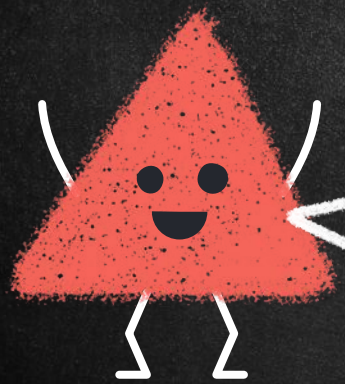
Let's do a survey exercise...

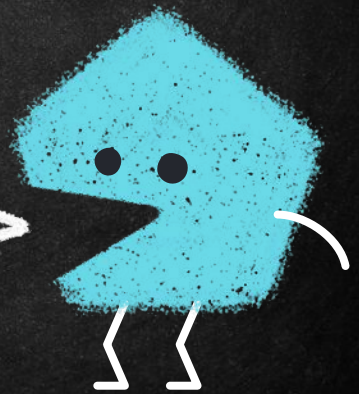
- Conduct a "survey session" to learn the product for 15 minutes
- Tour the product to map every important aspect of it

We will debrief together



Debrief





What is a PCO?

- A product coverage outline is a detailed, working reference used to answer questions such as:

What could we test?
What should we test?
What are we testing?
What did we test?
What did we NOT test?

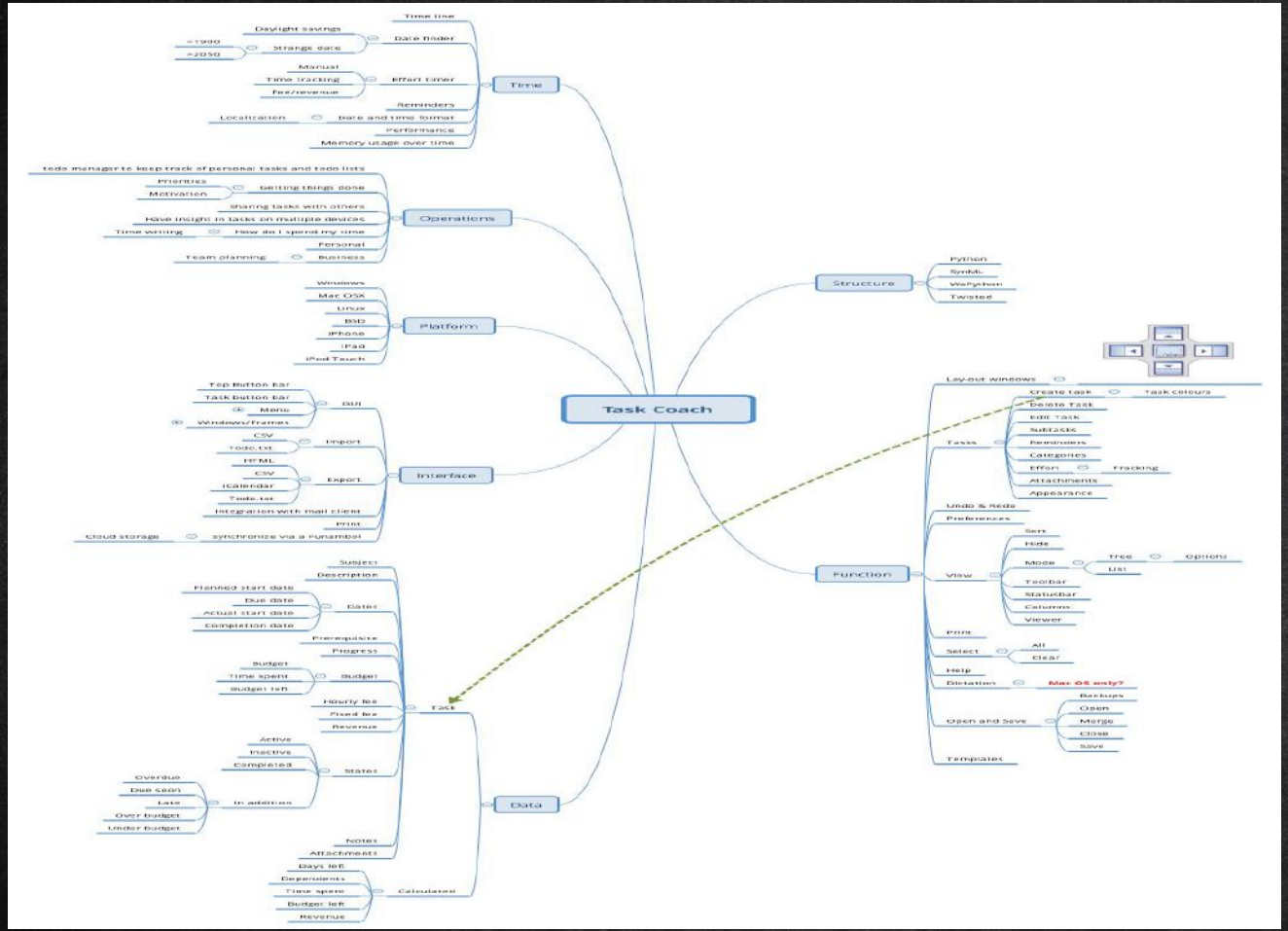
- It may be in plain text or in mindmap form. It may even be a diagram, or a set of Post-Its.

Product Coverage Outline

- Is an artifact (a map, list, diagram, sketch, table…) that identifies the factors or elements of a product that might be relevant to testing it
- Put these factors on the PCO if there is some chance you would otherwise forget or neglect them
- Factors include things users see, and things they don't see
- The Product Elements section of the Heuristic Test Strategy Model (SFDIPOT) provides a point of departure for creating a coverage outline

SFDIPOT

- Structure
- Function
- Data
- Interfaces
- Platform
- Operation
- Time



@huibschoots @alex_schl

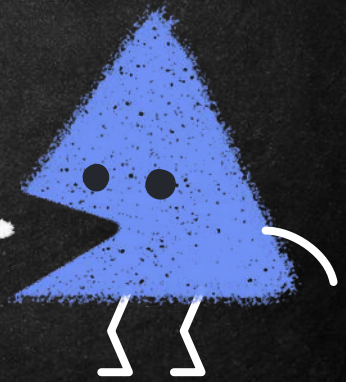
Watch the video

http://youtu.be/NUojNfDjIjw

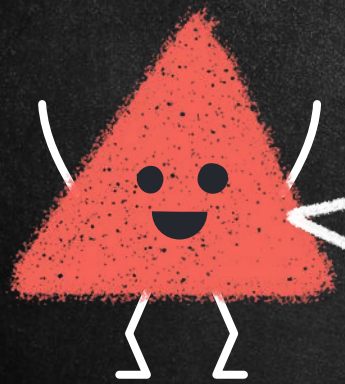
Let's do a survey exercise...

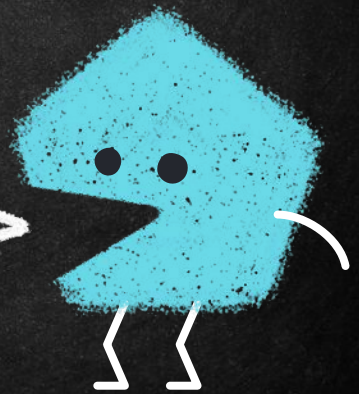
- Conduct another "survey session" to learn the product for 20 minutes
- Tour the product to map every important aspect
- Using a mind map, model an overview of the product
  (if you think of risks, write them down too)

We will debrief together

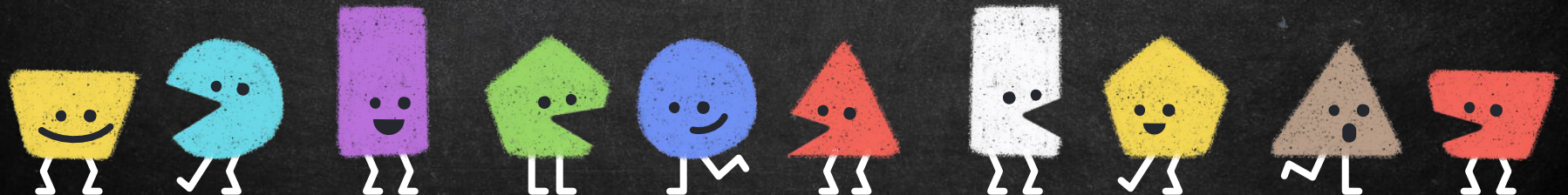


Debrief



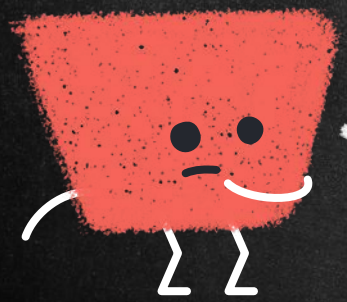


Sources & Heuristics



Sources & Heuristics

- How do we know what to test?
- How do we generate ideas?

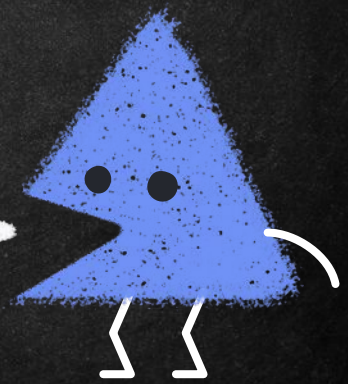


Exercise

- → What sources of information are there?
- → What information can we get from them?

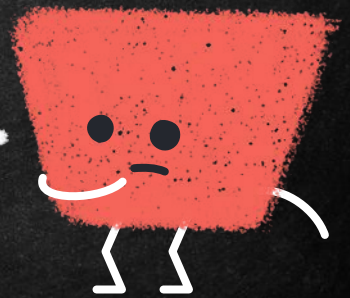
Discuss in a group for 15 minutes

We will debrief together



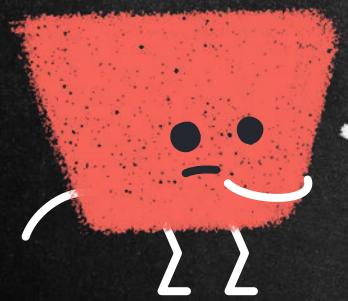
Heuristics

- A heuristic is "a fallible method for solving a problem or making a decision"
- A heuristic is a mental shortcut that allows people to solve problems and make judgements quickly and efficiently. These rule-of-thumb strategies shorten decision-making time and allow people to function without constantly stopping to think about their next course of action.
- Snap judgement / rule of thumb
- https://www.verywellmind.com/what-is-a-heuristic-2795235



Testing Heuristics

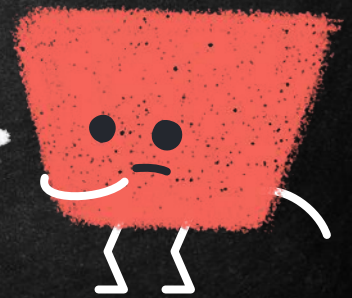
- FEW HICCUPS (Oracles: Consistency heuristics) http://www.developsense.com/blog/2012/07/few-hiccupps/
- Testing Heuristics Cheat Sheet (https://bit.ly/10r9Inn)
- Goldilocks (Elisabeth Hendrickson)
- Bad strings (htthttps://github.com/minimaxir/big-list-of-naughty-strings)
- Falsehoods programmers believe (https://spaceninja.com/2015/12/08/falsehoods-programmers-believe/)
- SFDIPOT - Heuristic for product elements
- RCRCRC for regression testing (http://karennicolejohnson.com/2009/11/a-heuristic-for-regression-testing)



RCRCRC:

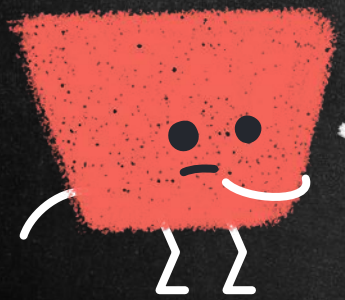
It is a regression testing heuristic

Recent: new features, new areas of code are more vulnerable
Core: essential functions must continue to work
Risk: some areas of an application pose more risk
Configuration sensitive: code that's dependent on environment settings can be vulnerable
Repaired: bug fixes can introduce new issues
Chronic: some areas in an application may be perpetually sensitive to breaking



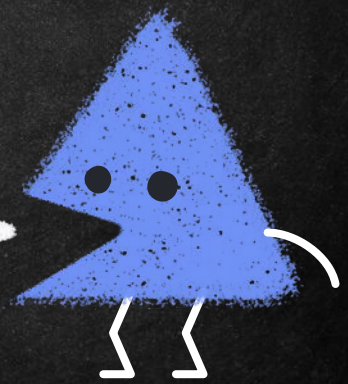
Some more...

- Personas
- Test Tours: Heuristics to guide exploratory testing http://www.developsense.com/blog/2009/04/of-testing-tours-and-dashboards/ and http://test-able.blogspot.com/2013/11/web-testing-exploratory-tours.html
- Software Quality Characteristics http://thetesteye.com/posters/TheTestEye_SoftwareQualityCharacteristics.pdf

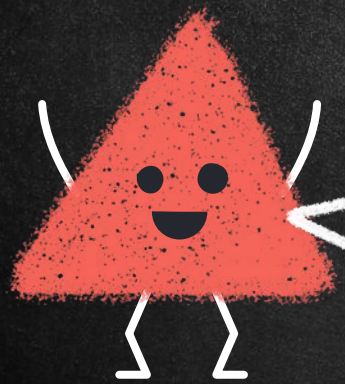


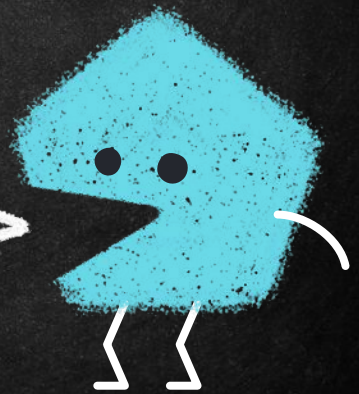
Exercise: using a heuristic

- Each choose a heuristic:
  - Cheat sheet
  - Tours
  - Quality Characteristics
- Find out what the heuristic is and how you could use it in your testing (15 min)
- Present your findings to the group (5 min)



Debrief

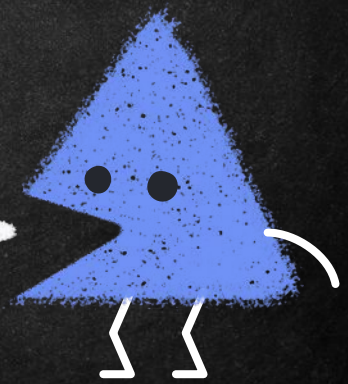




Exercise: discussion on testideas

→ How do you come up with testideas?

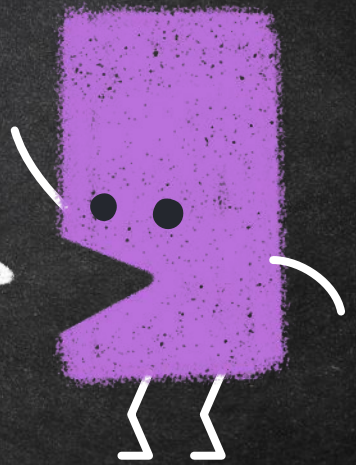
→ Do you use heuristics? Lists?



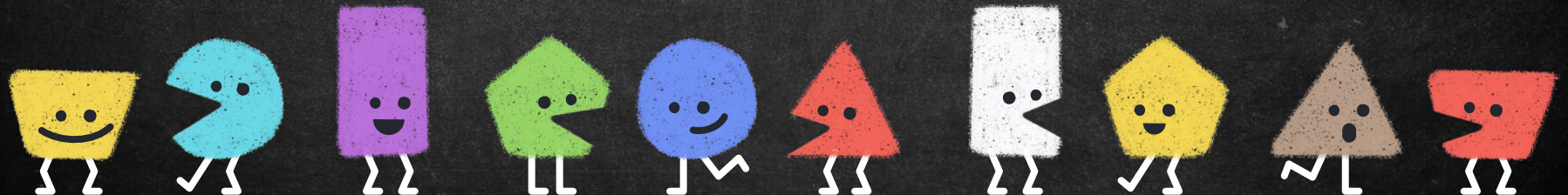
“

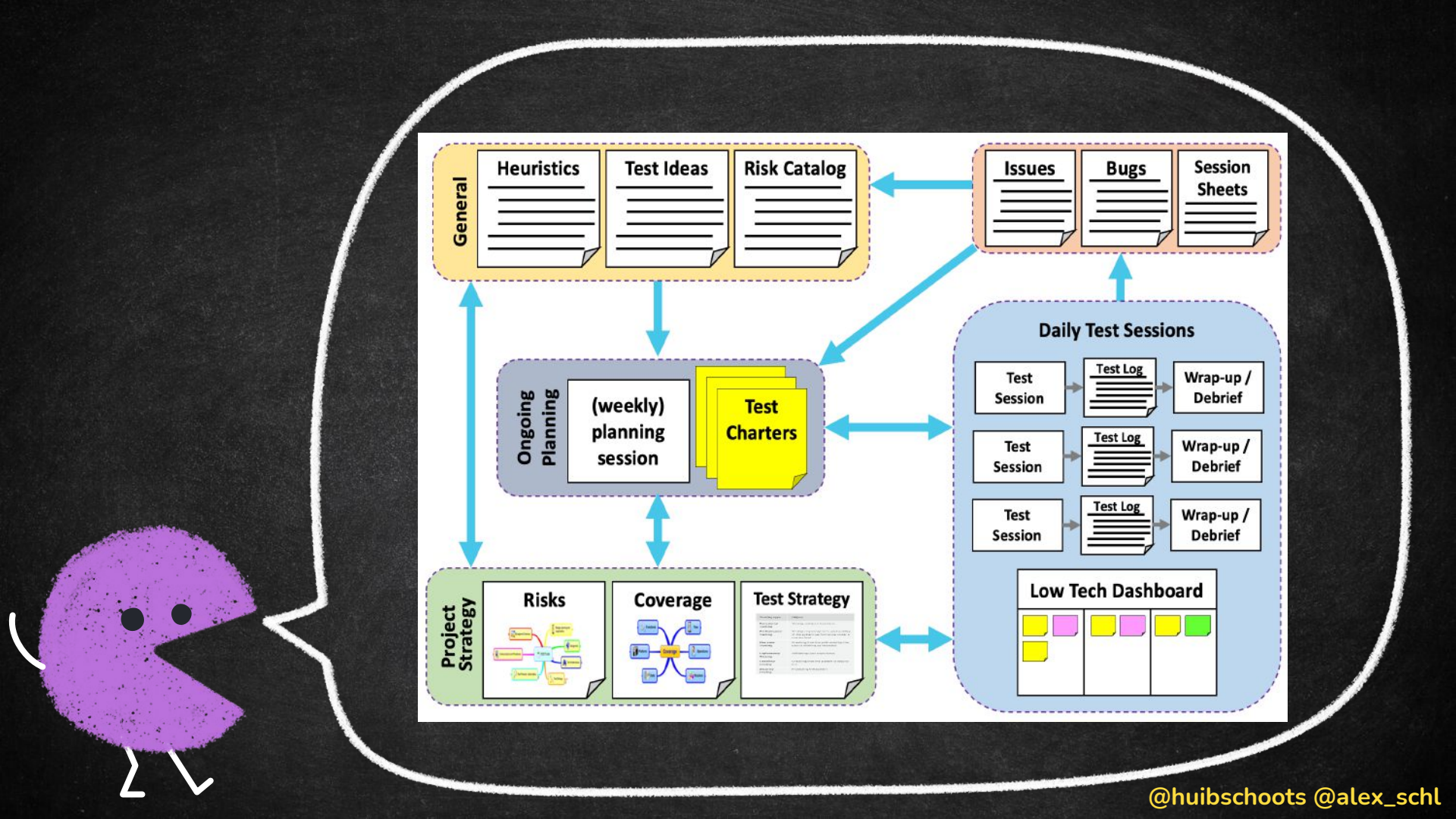
How to come up with Test Ideas?

http://erik.brickarp.se/2016/08/how-to-come-up-with-test-ideas.html



Session Based Test Management




General
Heuristics
Test Ideas
Risk Catalog
Issues
Bugs
Session Sheets
Daily Test Sessions
Ongoing Planning
(weekly) planning session
Test Charters
Test Session
Test Log
Wrap-up / Debrief
Test Session
Test Log
Wrap-up / Debrief
Test Session
Test Log
Wrap-up / Debrief
Project Strategy
Risks
Coverage
Test Strategy
Low Tech Dashboard

Test Charters

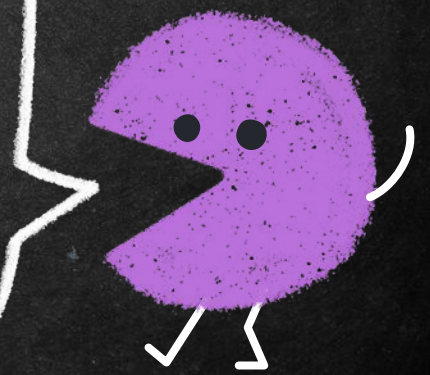
A charter is a one- to three-sentence mission for a testing session

Explore (target)
With (resources)
To discover (information)

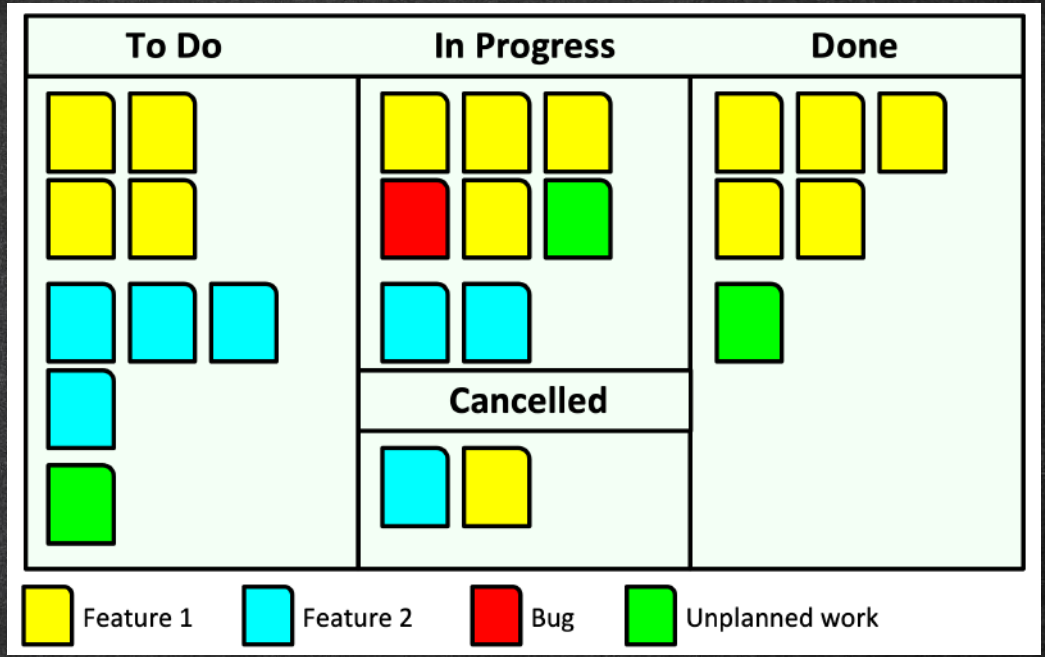
Explore Application X import menu. Identify important features with the goal of developing a coverage outline and a risk list.

My mission is to test <insert risk here> to <insert coverage here>

Read Chapter 4 of the product specification. Prepare a mind map, and discuss it with Peter (programmer) and David (architect).



Testing Dashboard



Charter Patterns: Evolving test strategy

Intake Sessions (Goal: negotiate mission)

→ "Interview the project manager about testing Xmind."

Survey Sessions (Goal: learn product)

→ "Familiarize yourself with Xmind."

Setup Sessions (Goal: create testing infrastructure)

→ "Develop a library of mindmaps for testing Xmind."

Analysis Sessions (Goal: get ideas for deep coverage)

→ "Identify the primary functions of Xmind."

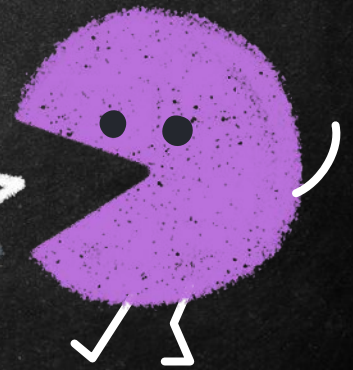
→ "Construct a product coverage outline."

→ "Brainstorm test ideas."

→ "Prepare a state model for state-based testing."

→ "Perform a component risk-analysis to guide further testing."

→ "Discover all the error messages in Xmind."



Charter Patterns: Evolving test strategy

Deep Coverage Sessions (Goal: find the right bugs)

- → "Perform scenario testing based on the scenario playbook."
- → "Perform a tour that achieves double-transition state coverage."
- → "Perform steeplechase boundary testing on the major data items."
- → "Test each error message in Xmind."
- → "Perform a function tour using the 2300 node mindmap."

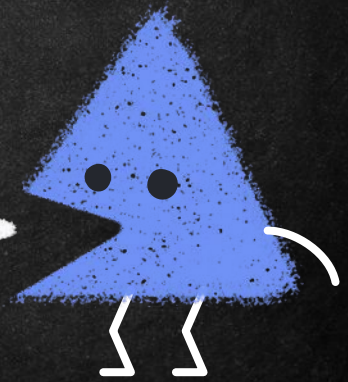
Closure Sessions (Goal: get ready to release)

- → "Verify the latest fixes."
- → "Re-test tutorial with the latest build."
- → "Review help files and readme."
- → "Go over deferred bugs with Customer Support people."
- → "Perform clean-machine install test."

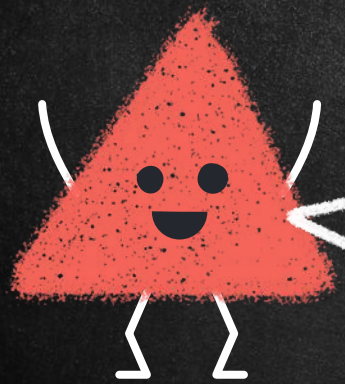
Exercise: create charters

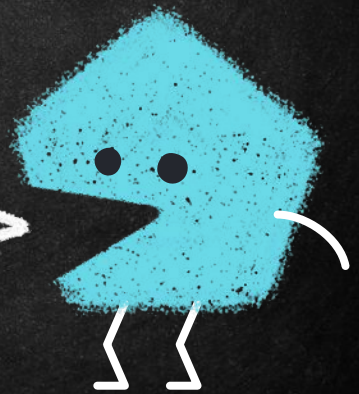
- Create a list of at least 5 charters for our test application:
- http://demowebshop.tricentis.com/

We will debrief together



Debrief



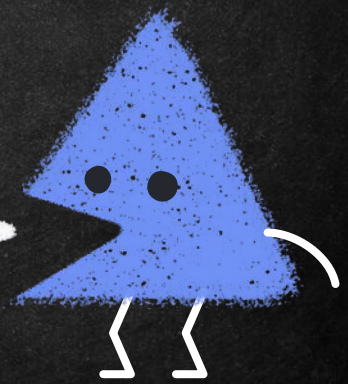


Exercise: test a charter

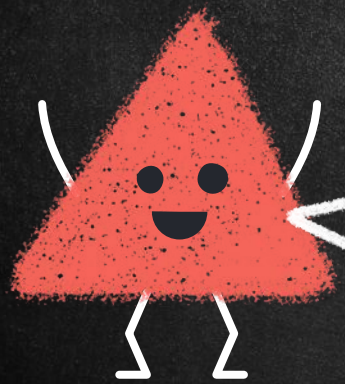
→ Select a charter for deep testing (30 minutes)

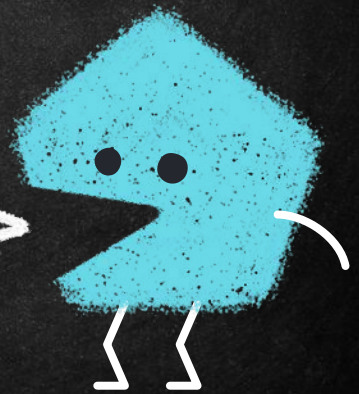
→ Execute a deep testing session on this charter

We will debrief together



Debrief





A Debrief heuristic: PROOF

Past: What happened during the session?

Results: What was achieved during the session?

Outlook: What still needs to be done?

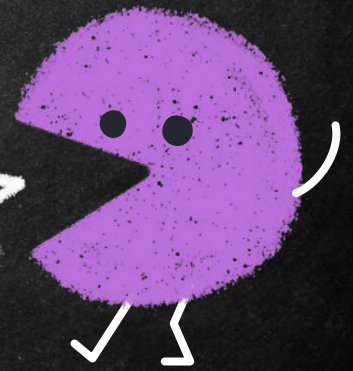
Obstacles: What got in the way of good testing?

Feelings: How does the tester feel about all this?

→ Discuss session and ask questions: new sessions may be chartered

→ Discuss session sheet to assure that both understand it

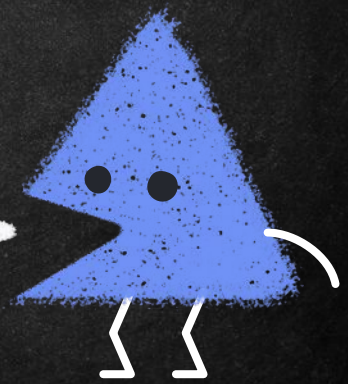
→ Coach & learn!



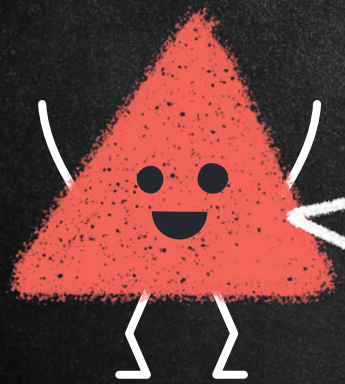
Exercise: Debrief your session

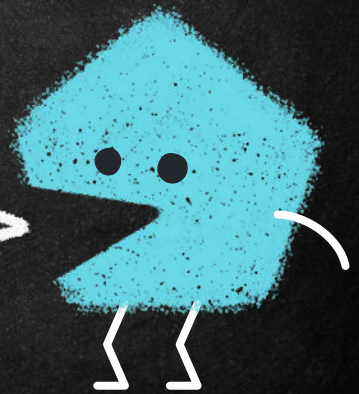
→ Do a debrief of the last session and use PROOF

Past:	What happened during the session?
Results:	What was achieved during the session?
Outlook:	What still needs to be done?
Obstacles:	What got in the way of good testing?
Feelings:	How does the tester feel about all this?



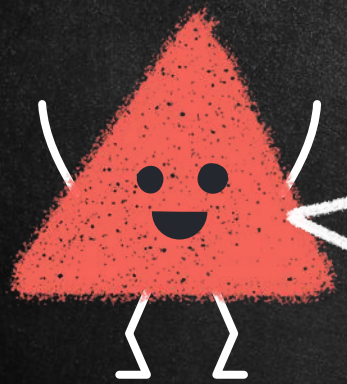
Debrief

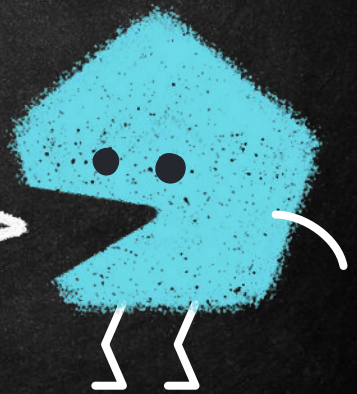




Wrap-up

Let's wrap this thing up!





Why ET is powerful

- Do what really needs to be done
- It facilitates experimentation, serendipity, discovery and learning
- It finds more bugs
- Create engagement: help people use their brains
- Take advantage of tacit knowledge and skill
- Use insights from experiments to inform the next
- Using the full creative power of exploration

Challenges?

- Exploratory Testing = (like testing in general) is not easy and needs (a lot) training and practice

- Often seen as most difficult:
  - Note taking
  - Test ideas
  - Coverage reporting
  - Managing ET

How to master ET?

- Just do it!
- Practice, practice and practice some more...
- Pair with colleagues
- Debrief & retrospect
- Testing Dojo
- TestOpsy
- Train creating test ideas fast
- Learn more about observation, experiments, biases, social science, ...

"Testing is about questioning & learning under conditions of fundamental uncertainty."

RST

"A tester knows that things can be different"

Jerry Weinberg

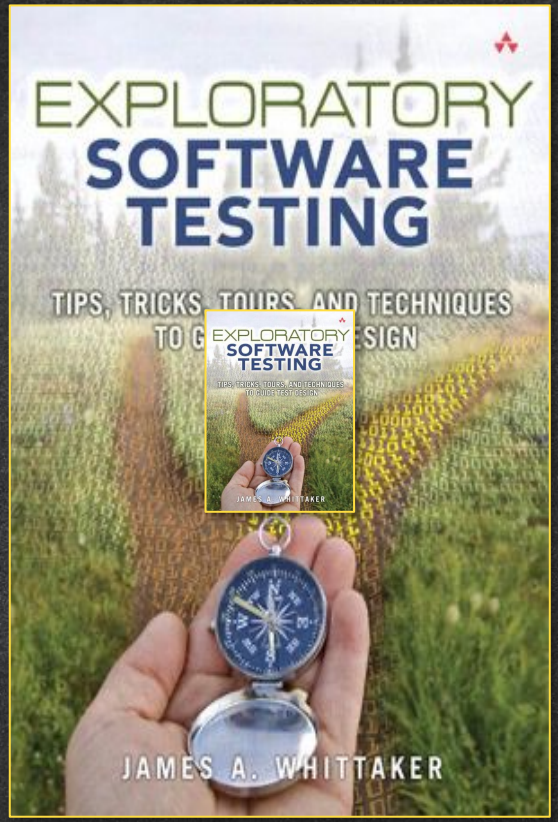
"I've participated and organized many testing competitions. I've never seen someone win such a competition by writing down tests. Doesn't that tell us something?"

James Bach

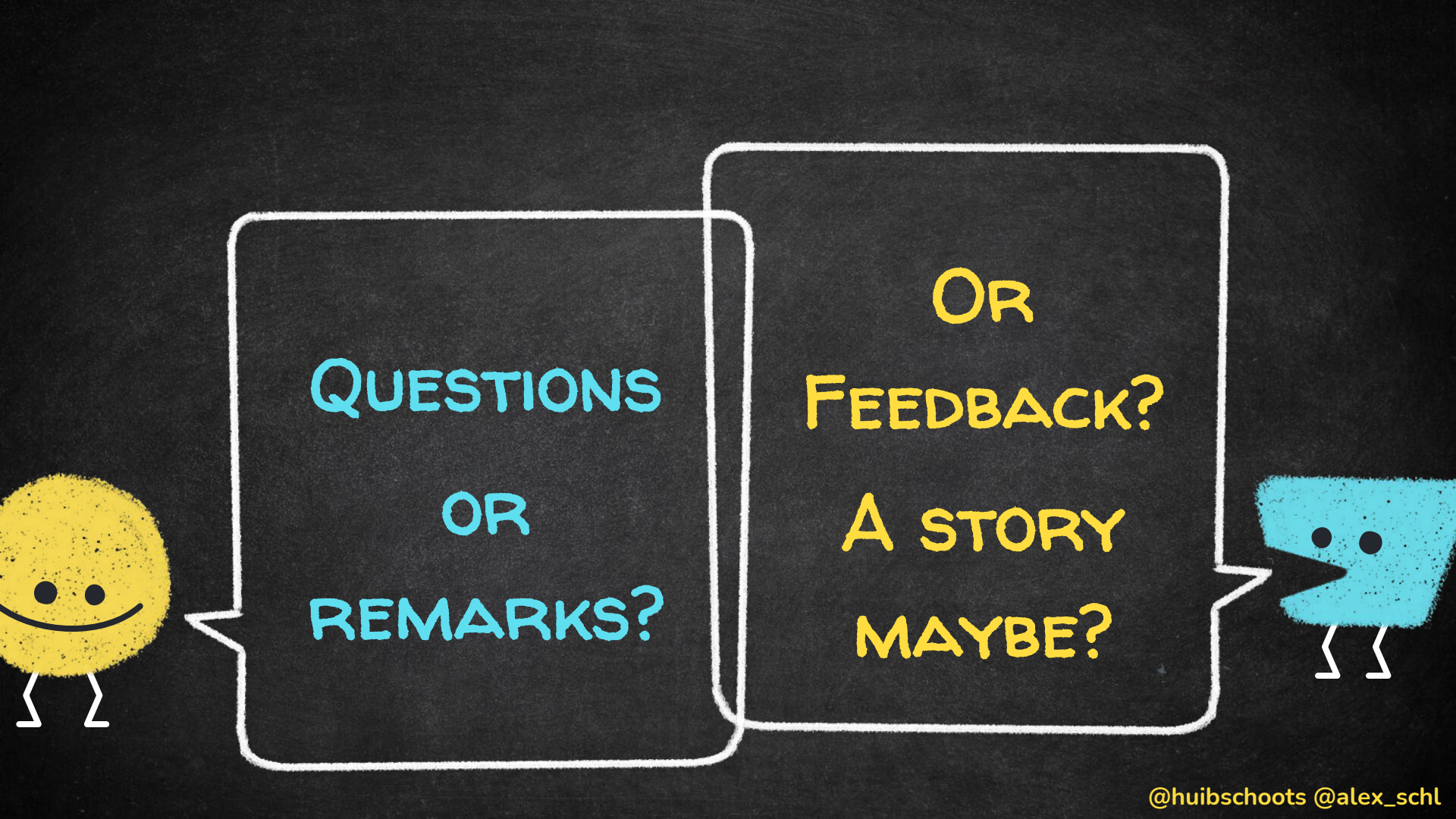
If you cannot trust your testers, you do not make them write more detailed test case. But you train them!

Rikard Edgren – EuroStar 2012
Gitte Ottosen – ATD 2012

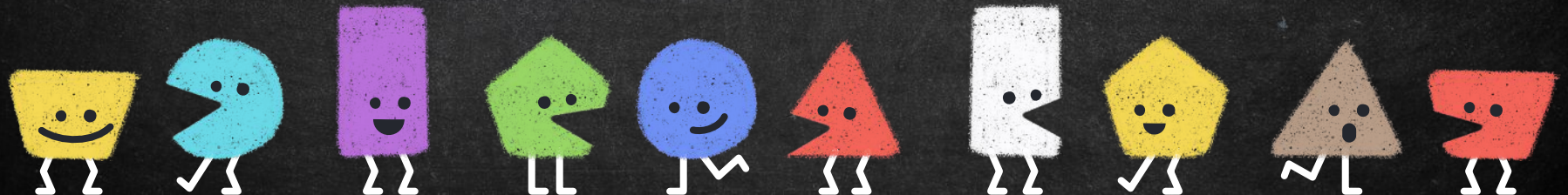

The Pragmatic Programmers
Explore It!
Reduce Risk and Increase Confidence with Exploratory Testing
Elisabeth Hendrickson
Edited by Jacquelyn Carter


EXPLORATORY
SOFTWARE TESTING
JAMES A. WHITTAKER

Value


Questions
or
remarks?
Or
Feedback?
A story
maybe?
@huibschoots @alex_schl

Thank you!



HUIB@QUALITYACCELEARATORS.NL

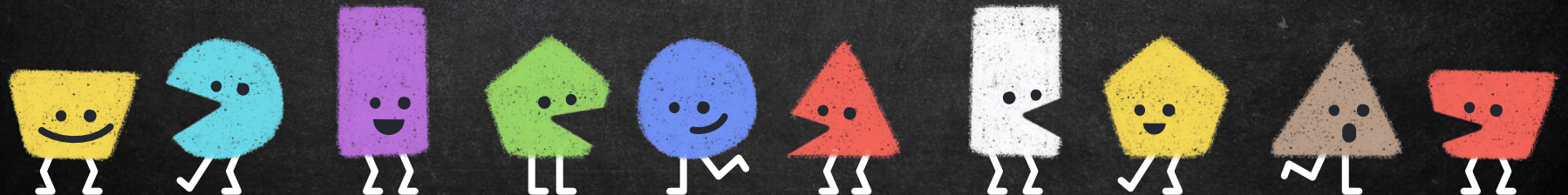
@HUIBSCHOOTS

HTTP://WWW.HUIBSCHOOTS.NL

ALEX.SCHLADEBECK@BREDEX.DE

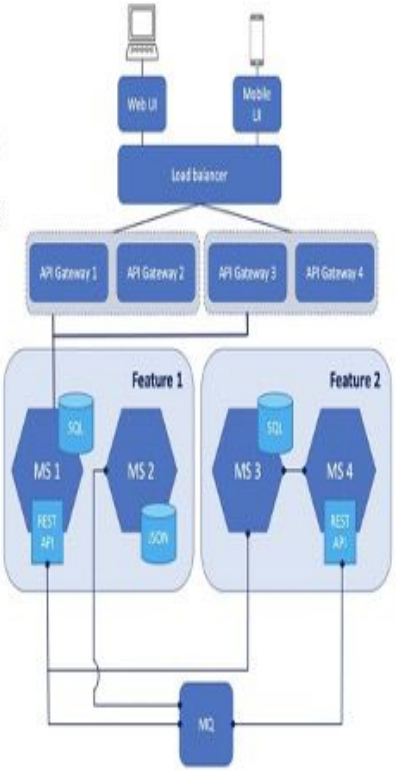
@ALEX_SCHL

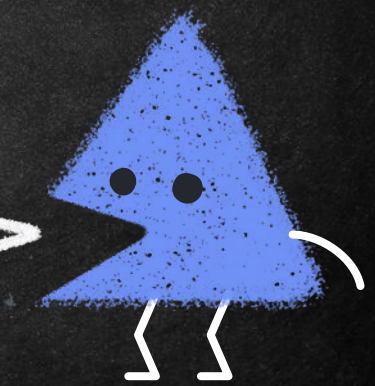
HTTP://WWW.SCHLADEBECK.DE



Things to ask during Risk Analysis

- [pointing at a figure in a diagram] What if this function doesn't work?
- Can this function ever be invoked at the wrong time?
- [pointing at any part of the diagram] What error check are we doing here?
- [pointing at an arrow] What exactly does this arrow mean? What happens if it doesn't work?
- [pointing at a data stream] If the data going from here to there is somehow corrupted, how do we see it? What happens then?
- What is the greatest load this process can handle?
- On which external components, services, configurations does this process depend?
- Can the sources or components drawn here be influenced by another process? How can I crash the process?
- Is this the whole story? What did you leave out?
- How do you test this while you are building it?
- What are you most worried about? What do you think we need to test?





References & more info

- Explore it (book) by Elisabeth Hendrickson
- Applying Session-Based Test Management by Djuka Selendic https://www.infoq.com/articles/session-based-test-management/
- "Evolving Understanding of Exploratory Testing" en "Structures of Exploratory Testing" http://www.developsense.com/resources.html
- Test cases are not testing: towards a culture of test performance by James Bach & Aaron Hodder https://www.satisfice.com/download/test-cases-are-not-testing
- Considerations when testing a software application in a context-driven way https://www.huibschoots.nl/wordpress/?p=2781
- Little black book on Test Design http://www.thetesteye.com/papers/TheLittleBlackBookOnTestDesign.pdf
- How to come up with test ideas by Erik Brickarp http://erik.brickarp.se/2016/08/how-to-come-up-with-test-ideas.html
- Microheuristics by Alex Schladebeck – https://schladebeck.de/microheuristics
- Test Insane – Testing Mind maps http://apps.testinsane.com/mindmaps/
- Rapid Software Testing https://rapid-software-testing.com/
- Experience report on using a Product Coverage Outline http://prairietester.blogspot.nl/2013/09/monday-product-coverage-outlines.html
- Testing Story http://www.developsense.com/blog/2012/02/braiding-the-stories/
- Collected resources: https://www.huibschoots.nl/links

Template for slides by:

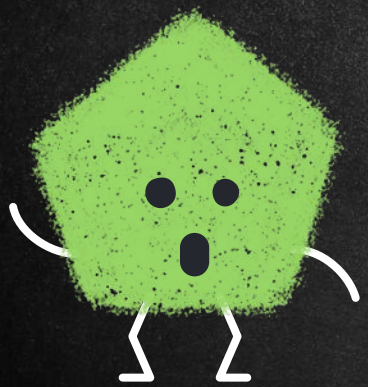
Slides Carnival

@huibschoots @alex_schl

More References & even more info

TestOpsy:

- Exploratory Testing Skills & Dynamics (in RST Appendices) –
- Skills mind map – http://goo.gl/VCQOIN
- Webinar Testopsies Dissecting Your Testing – https://youtu.be/dz26_LUt2Wk
- Report of a Testopsy – http://patternsofproof.wordpress.com/2015/03/07/on-performing-an-autopsy/
- Report of a Testopsies workshop – http://www.brendanconnolly.net/testopsies/



AGILE TESTING DAYS USA