2008

Software Testing Basics

Thomas Klein

Welcome.

Name: Thomas Klein

Age: 42, from: Cologne, Germany

Started w/ professional developing in 1992 (COBOL, VB)

Software QA since 1998

Focus on TelCo, mainly Deutsche Telekom, Vodafone D2

Track and Team lead in on-/offshored QA projects (Sapient corp.)

Agenda

1 - Introduction

- What is testing?
- Flavors of testing

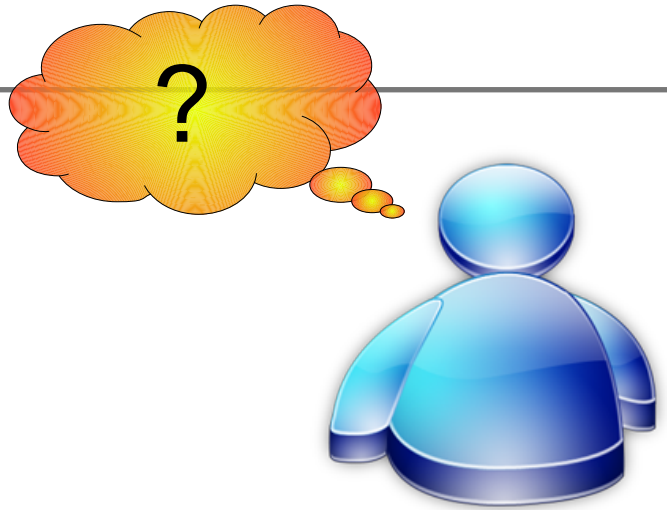
2 - Supporting your eCS developer

- Hands on: What can be tested?
- How-to essentials: Test cases
- If something unexpected happens...
- Bug reporting (...Do's and Dont's)

3 - Working together

- How developers can contribute
- Helpful things for collaboration

1 / 3 - Introduction



What is testing

- Definitions
- ...and what does that mean?

Flavors of testing

- Commonly known buzzwords
- Real life: Commercial vs. Community-based

What is testing? Definitions.

> *The process of operating a system or component*
> *- under specified conditions*
> *- observing or recording the results and*
> *- making an evaluation of some aspects of the system or component.*

ANSI/IEEE Std. 610.12-1990

> *The verifiable and always repeatable proof of correctness of a software component in relation to requirements defined upfront.*

according to Ernst Denert, “Software-Engineering”

...and what does that mean?

- Having predefined “requirements” (→ doc, manual, …)
- Deriving expectations (“test cases”) from requirements
- Having a “known” environment (defined, agreed, …)
- Using the software :)
- Observing behaviour
- Recording results
- Comparing to expectations
- Something repeatable / reproducible

...and what does that mean?

- Having predefined “requirements” (→ doc, manual, …)
- Deriving expectations (“test cases”) from requirements
- Having a “known” environment (defined, agreed, …)
- Using the software :)
- Observing behaviour
- Recording results
- Comparing to expectations
- Something repeatable / reproducible

Flavors of testing: There you go.

Flavors of testing: Commercial vs. Community

In the commercial world, in large projects:

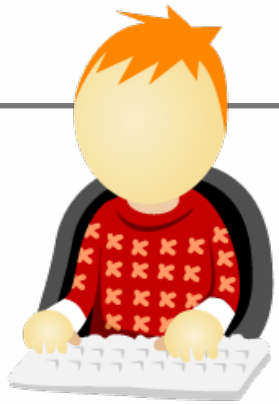
- Wealth of tools and methodologies
- Constraints: Time, Resource, ...Money
- Things need to be plannable, trackable, manageable and affordable
- Multi-stage as per lifecycle, scope, requirements (SOX, SLA, customer...)
- “Agile vs. waterfall” - “Manual vs. automated” ...**but planned**

In the FOSS / community-driven world:

(can be made commercial-level, depends on scope, resources, skills... management?)

- In small projects often 1:1 and/or no tools
- No time/money constraints but unreliable resources (if at all)
- Requirements? `Test stages?
- Waterfall & Manual ...and mostly **ad-hoc?**
- “QA” vs. “Testing” vs. “Bug Reporting” – how much do we need?

2 / 3 - Supporting your eCS developer



- **Hands on: What can be tested**

  Start somewhere.

- **How-to essentials: Test cases**

  What's that and why does it matter?

- **If something unexpected happens...**

  A checklist

- **Bug reporting: DO's and DON'Ts**

  How to make friends in a minute

Hands on: What can be tested?

- The “package”: Download, Media...
- Installation: System requirements, OS, Versions...
- Migration and Upgrade scenarios
- The paperwork: Manuals, readmes...
- Localization (what about DBCS?)
- Functional: Core features, actual processing, “verify and try to break it”
- Useability: Browsers, resolutions, “mouse vs. keyboard”, ...
- Interfaces: Scripting, Plugins, named pipes etc.
- Performance, Load (huge input), Stress (low resources)

How-to essentials:

Stephan Assmus
(Haiku project team)

> *Unfortunately, we have at least one remaining file system bug that corrupts data. We need this one fixed and* ***hopefully we can find a reliable test case to reproduce it****.*

How-to essentials: Test case example

Test Case #1234 – Check Login behaviour if user browser doesn't accept cookies

Preconditions:
Apache up and running, User credentials for a valid account, cookies turned off in Browser
Test data used:

Step 01:
Action: Invoke Landing page in browser (www.myserver.com)
Expected: Landing page appears and has
- UserID entry field (empty)
- Password entry field (empty)
- “login” button (red)

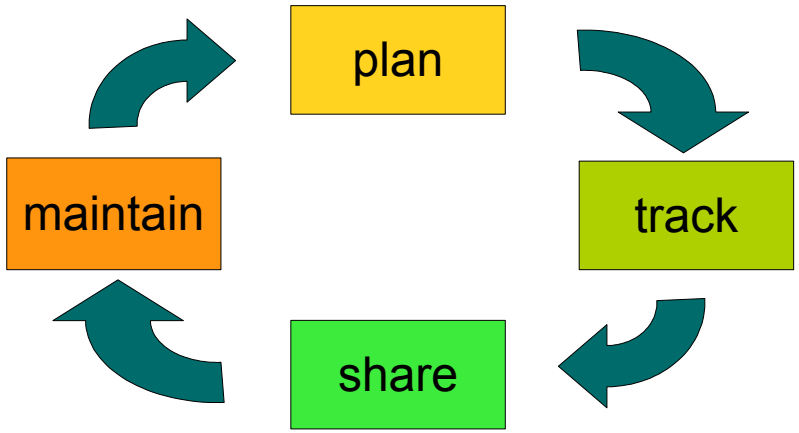
Actual:

Step 02:
Fill in credentials for test account, click “login button”
Expected: Message in browser “unable to login - please turn on cookie support” or similar
Actual:

How-to essentials: Test cases

Get acquainted with test cases, as they...

- are the core work and basis for the “remaining QA stuff”
- must be understandable and reproducible ...also by others
- are the “work” link between developers and testers (reviews...)
- as by the quote... are sometimes also important for developers ;-)



Testers need attitude of thoroughness and elaborateness...

...and guts and grit.

If something unexpected happens...

Before reporting an issue, ask yourself:

- Is that really an error?
- Are you running the correct version? S/W requirements matched?
- Is that a “known error” (see readme/manual)?
- Is that error already submitted (ticket tool, list of bugs)?

When it's likely to be an issue:

- Can you reproduce it?
- Takes notes of what might be important for reproducing
- Dump or describe test data used
- Can you log / take screenshots ?
- What is the impact? Workaround possible?
- Eventually: Cross-check other system / software version

Bug reporting: DO's and DON'Ts

Make sure to...

- mention version used, environment, configuration, etc.
- describe what happened (step by step, actions...)
- tell expected and actual behaviour (unless obvious)
- attach whatever might be useful
- In “planned” testing: Impact analysis (blocker?)

And avoid...

- **to code in your report. You might see the error but not the big picture.**
- **to assume.** Either prove or show it ...or forget it.
- **to suggest something (u**nless being asked). Use feature request.
- **to “metoo”.** Unless you can add information or were asked

Bug reporting: DO's and DON'Ts

One bug per ticket please

- Don't put multiple issues in one ticket (unless it's a mailed report)
- Each of the issues might get a different reply
- Tedious to track, forward
- “Large part” will block delivery of small fixes
- However: Group similar symptoms (GUI, typos, ...)
- Use ticket linkage where useful to show bug dependencies

Avoid ping-pong

- Learn from the reasons
- Open new or different ticket(s)

3 / 3 – Working together

How developers can contribute

- Unleash your evil alter ego...

Helpful things for collaboration

- Who's doing what? And where/why/when?

How developers can contribute

Sometimes, think like a PM

- Care about processes (milestones, deadlines, scoping, managing)
- Tell what you need in testing (functional, environmental, ...)
- ...and beyond SW testing (docs, web page, localizing)
- Provide bug tracker tools or e-mail templates
- Share (and track) your bugs and fixes (-> changelog, history)

...and sometimes like a tester

- You need specific info? Let a tool grab it.
- What had happened: “/debug” is your friend
- Make submitting bugs easy ***for testers*...**

Helpful things for collaboration

Ease contribution, motivate, manage: Tools!

- Bugtracker (Mantis, Flyspray, bugzilla, …)
- Wiki (Manual, Docs, FAQ, How-To, Planning...)
- Newsletters, Reports, …
- Have POCs for ramping up “the noobs”

Show that you care

- Ask suggesters to become “testing sponsors” / “Subject Matter Experts”
- Give credits
- Delegate, talk, plan

closure

- Questions?
- Was it as good for you as it was for me?

Thank you!