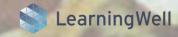
Good Testers Are Often Lucky

using serendipity in software testing

Nordic Testing Days, 6 June 2014
Rikard Edgren



Serendipitous Life

- Childhood
- Friends
- Education
- Job
- Family

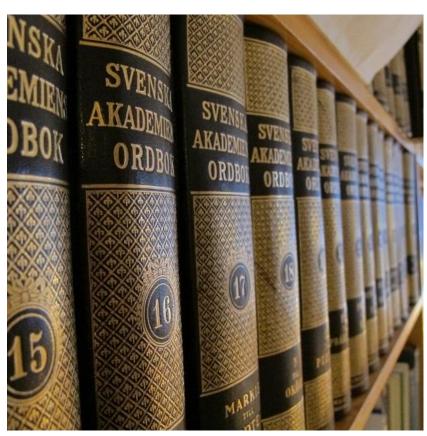


It's about creating opportunities for good luck



Serendipity Definition

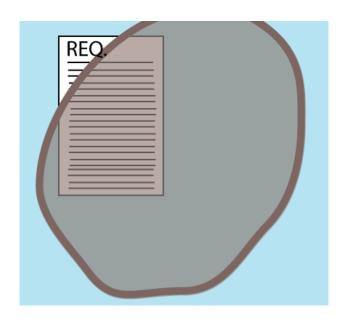
- "finding something valuable when looking for something else, thanks to an observant mind
- Serendipitet
- Serendipität
- Serendipia
- Sérendipité
- Serendipo
- Serendipisyys
- セレンディピティ
- سرنديپيتى 🕨
- Juhuslik avastus



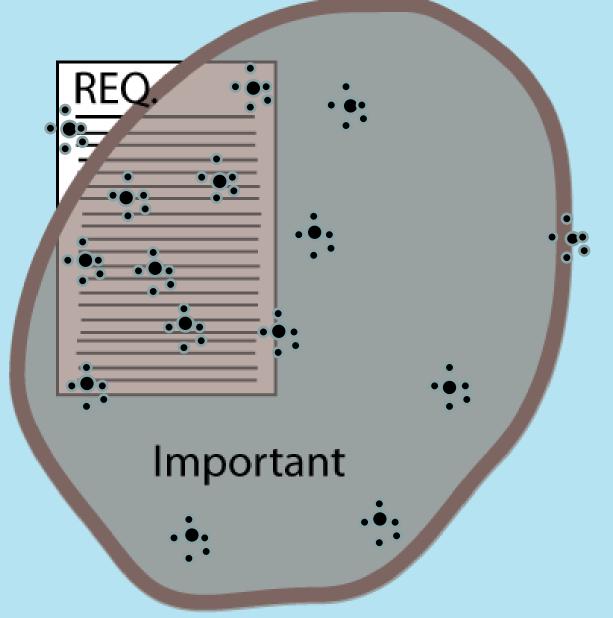


Sampling & Serendipity

- We can't test everything, we sample.
- We can observe carefully.
- We want to make many, rich tests.
- We change sampling strategy as we learn more.









Everything

Prepare for Serendipity

- **▶** Error-Prone Machine
- Background Complexity Heuristic





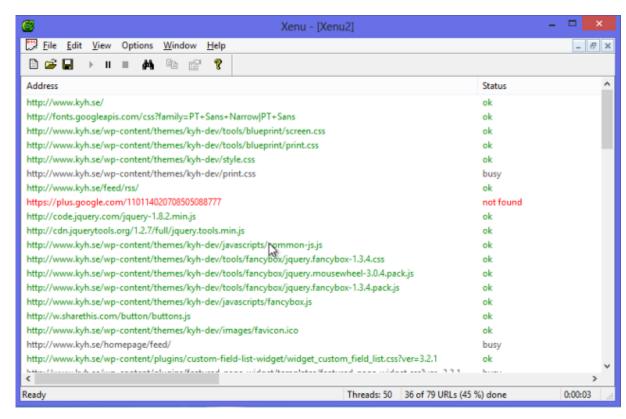
Vary for Serendipity

- Don't run the same test twice (unless you really have to)
- Observe carefully, on many places
- Do One More Thing Heuristic
- Galumphing
- Also vary your testing techniques and approaches (Lessons Learned 283)



Serendipity Observation Examples

- Look carefully
- Notepad Heuristic
- Visualizing





Serendipity Observation Examples

- Look carefully
- Notepad Heuristic
- Visualizing





Ongoing Serendipity

- Things we know always matter, can be tested "for free"
- Make tests richer.





and support? Software Quality Charact **Ongoing Serendipity** is, and handle different environments or missing components. vith appropriate footprint. Capability. Can the product perform valuable fun f configuration and settings. ther resources removed when uninstalling? Completeness: all important functions wanted by end users are Accuracy: any output or calculation in the product is correct and presented with significant digit: Efficiency: performs its actions in an efficient manner (without doing what it's not s Interoperability: different features interact with each other in the best way. IT-bility. Is the product easy to install, maintain and support? Concurrency: ability to perform multiple parallel tasks, and run at the same time as - Data agnosticism: supports all possible data formats, and handles noise. Reli ne product can run on intended operating system versions, and follows typical behavior. **Capability.** Can the product perform valuable functions? oduct, and its data, works with other applications customers are likely to use. uct's ability to blend in with configurations of the environment. **Compatibility.** How well does the product interact with software and environments? **Reliability.** Can you trust the product in many and difficult situations? Learnability: it is fast and easy to learn how to use the product. - Memorability: once you have learnt how to do something you don't for Discoverability: the product's information and capabilities can be discoverable. **Supportability.** Can customers' usage and problems be supported? Operability: an experienced user can perform common actions very fas - Interactivity: the product has easy-to-understand states and possibility Control: the user should feel in control over the proceedings of the software. Traceability: the product logs actions at appropriate levels and in usable format. or doubt? - Controllability: ability to independently set states, objects or variables. **Usability.** *Is the product easy to use?* - Observability: ability to observe things that should be tested. - Monitorability: can the product give hints on what/how it is doing? - Isolateability: ability to test a part by itself. Documentation: there is a Help that helps, and matches the functionality. **Testability.** *Is it easy to check and test the product?* Charisma. Does the product have "it"? - Uniqueness: the product is distinguishable and has something no one else has. Maintainability. Can the product be maintained and extended at low cost? - Flexibility: the ability to change the product as required by customers. **Charisma.** Does the product have "it"? - Extensibility: will it be easy to add features in the future? - Simplicity: the code is not more complex than needed, and does not obscure test design, execution and evaluation. Readability: the code is adequately documented and easy to read and understand. Hype: should the product use the latest and great Expectancy: the product exceeds expectations at **Maintainability.** Can the product be maintained and extended at low cost? Attitude: do the product and its information hav Directness: are (first) impressions impressive? Story: are there compelling stories about the product's inception, construction Portability. Is transferring of the product to different environments enabled? Security. Does the product protect against unwanted usage? support a different environment? **Security.** Does the product protect against unwanted usage? nmon interfaces or official standards? Secrecy: the product should under no circumstances disclose information about the underlying systems. User Interface-robustness: will the product look equally good when translated? - Invulnerability: ability to withstand penetration attempts. - Virus-free: product will not transport virus, or appear as one. **Portability.** *Is transferring of the product to* Piracy Resistance: no possibility to illegally copy and distribute the software or code - Compliance: security standards the product adheres to Performance. Is the product fast enough? different environments and languages enabled? - Capacity: the many limits of the product, for different circumstances (e.g. slow network.) - Resource Utilization: appropriate usage of memory, storage and other resources. **Performance.** *Is the product fast enough?*

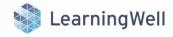
 Feedback: is the feedback from the system on user actions appropriate? - Scalability: how well does the product scale up, out or down?

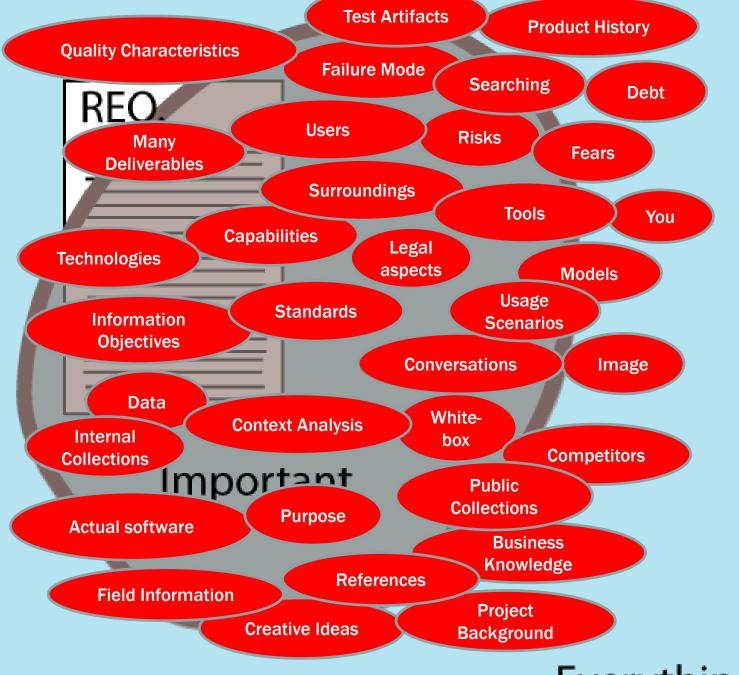
http://thetesteve.com/posters/TheTestEve SoftwareQualityCharacteristics.pdf

Connect for Serendipity

- Daniel Liestman wrote an article about serendipity for library research.
 - who wants to admit they found it by chance?

- Perseverance thoroughness and hard work
 - Look often, and at many places, use variations.
- Altamirage tacit knowledge
 - Hidden heuristics and invisible skills
- Sagacity ability to make good judgments
 - Connect observations and experience, the more you know, the better...





Everything

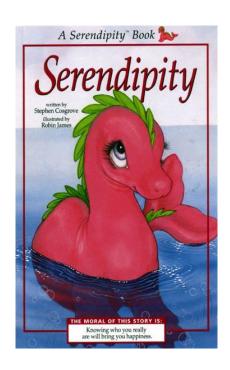
Serendipity Quotes

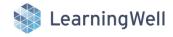
- you can see lot by just looking (Yogi Berra)
- this serendipity is what makes doing qualitative research and analysis so much fun (Strauss/Corbin)
- chance favors the prepared mind (Pasteur)
- rely less on top-down planning and focus on maximum tinkering and recognizing opportunities (Taleb)
- computers are marvellous, but they suck at serendipity (Edgren)
- wouldn't it be interesting to... (your next great idea)



Serendipity Summary

- Software testing is oozing with serendipity.
- Serendipity can be your friend and rescue, don't hide it.
- Learn a lot, prepare, do many tests and observe!

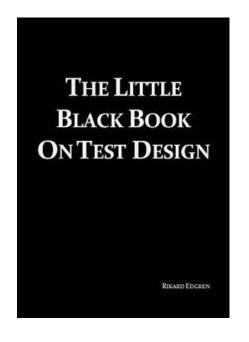




Questions

???

- Further reading:
 - Liestman, Chance in the Midst of Design: Approaches to Library Research Serendipity
 - Edgren/Emilsson/Jansson, Software Quality Characteristics http://thetesteye.com/posters/TheTesteEye_SoftwareQualityCharacteristics.pdf
 - Edgren/Emilsson/Jansson, 37 Sources for Test Ideas http://thetesteye.com/posters/TheTestEye_SourcesForTestIdeas.pdf
 - Edgren: The Little Black Book on Test Design http://thetesteye.com/papers/TheLittleBlackBookOnTestDesign.pdf



www.thetesteye.com

rikard.edgren@thetesteye.com

