Software Testing Heuristics & Mnemonics

Karen N. Johnson

Session Description

Are you curious about heuristics and how to use them?

In this session, Karen Johnson explains what a heuristic is, what a mnemonic is, and how heuristics and mnemonics are sometimes used together.

A number of both heuristics and mnemonics have been created in the software testing community. Karen reviews several of each and gives examples of how to use and apply heuristics and mnemonics.

In this session, Karen outlines how to create your own mnemonics and heuristics. She shares ways to use both during Exploratory Testing.

Karen N. Johnson

- Software Test Consultant
- Published Author
- Teach Software Testing
- Speak at conferences
- Co-founder of WREST, the Workshop on Regulated Software Testing
- Website: www.karennjohnson.com
- Twitter: @karennjohnson

What is a heuristic?

Heuristic

(/hjʉˈrɪstɨk/; or /hyoo-ris-tik/;

Greek: "Εὑρίσκω", "find" or "discover")

Refers to experience-based techniques for problem solving, learning, and discovery. Where an exhaustive search is impractical, heuristic methods are used to speed up the process of finding a satisfactory solution. Examples of this method include using a rule of thumb, an educated guess, an intuitive judgment, or common sense.

Heuristic Examples
(not from software testing)

From George Polya's book: "How to Solve It"

- If you are having difficulty understanding a problem, try drawing a picture.
- If you can't find a solution, try assuming that you have a solution and seeing what you can derive from that ("working backward").
- If the problem is abstract, try examining a concrete example.

What is a mnemonic?

Definition of mnemonic:

a device (such as a rhyme or acronym) used as a memory aid

Example:

To memorize the North American Great Lakes: the acronym HOMES - matching the letters of the five lakes (Huron, Ontario, Michigan, Erie, and Superior)

Heuristics do not have to have a mnemonic

- A well-known heuristic and an example of a heuristic that does not have a mnemonic is Elisabeth Hendrickson's "Goldilocks."
- Elisabeth highlights the following: Too big, too small and just right.
- By recalling "Goldilocks" I think about data entry fields and other variables where I can test with too large of an entry, too small of an entry and testing a more "expected" entry.
- Imagine trying to enter a date field with all 9's or a currency field with a negative number.

Let's try an example of how a heuristics & mnemonic can be used in testing.

Ready?

Sample regression mnemonic: RCRCRC

The mnemonic is RCRCRC used to remember the heuristic to help with regression testing. Each letter represents a word I use to help me discover testing ideas.

- Recent
- Core
- Risky
- Configuration
- Repaired
- Chronic

Sample regression mnemonic: RCRCRC

First letter: R for Recent

- Recent – meaning – what new features or new areas of code have been added?
- What changed recently?
- What testing around those changes should I think about?

Sample regression mnemonic: RCRCRC

Second letter: C for Core

- Core – meaning – what essential or critical functions or features must continue to work?
- What does this application primarily do?

Sample regression mnemonic: RCRCRC

Third letter: R for Risky

- Risk or Risky – meaning – what features or areas of code are inherently more risky?
- What features of the application do I want to safeguard?

Sample regression mnemonic: RCRCRC

Fourth letter: C for Configuration Sensitive

- Configuration sensitive – meaning – what code is dependent on environment settings?
- What testing do I want to plan based on the environment?
- This testing might include planning a subset of tests for production after go live.

Sample regression mnemonic: RCRCRC

Fifth letter: R for Repaired

- Repaired– meaning – what code has been changed to address defects and therefore may have created issues?
- What testing do I want to do based on defects fixed?

Sample regression mnemonic: RCRCRC

Sixth letter: C for Chronic

- Chronic – meaning some code may seem to be chronically breaking or having issues.
- This letter reminds me to test features that seem to break often.

How about another example?

I used James Bach's well-known heuristic SFDPOT and applied that heuristic to mobile testing.

SFDPOT

SFDPOT, heuristic by James Bach

S is for Structure

Test what it's made of.

How I applied this to mobile testing:

- Can I download the app?
- Can I download an update?
- Can I update the app when I have multiple apps to update?
- What happens when the operating system of the device is updated?
- Does the app write to an SD card?
- What if the card is filled, removed or changed?

SFDPOT

SFDPOT, heuristic by James Bach

F is for Function

Test what it does.

How I applied this to mobile testing:

- Does the app or site perform the tasks or features it was designed to?
- Does the app or site prevent, block or not include features not intended on a mobile device?
- Does the app or mobile website ever direct me to the “full site or desktop version”?
- Does the app direct me to turn on necessary related services such as location services if those services are shut off?

SFDPOT

SFDPOT, heuristic by James Bach

D is for Data

Test what it does it to.

How I applied this to mobile testing:

- Does the app find time related data (if relevant, such as movie show times) based on the device time?
- Does the app find locations based on my location (such as movie theaters or hotels)
- Does the app integrate with any hardware such as bar code scanners for payment or other purposes?
- If the app tracks rewards or points for frequent purchases, does this data get updated?
- If the app integrates with a desktop profile or account information (such as My Profile or My Account), does the app and the desktop version keep information in sync with each other?

SFDPOT

SFDPOT, heuristic by James Bach

P is for Platform

Test what it depends upon.

How I applied this to mobile testing:

- Does the app use location services?
- Does the app depend on any device settings?
- Does the app detect the platform it is running on such as a phone or tablet?
- Can I find the app in the app store?

SFDPOT

SFDPOT, heuristic by James Bach

O Is for Operations

Test how it's used.

How I applied this to mobile testing:

- Does the app function when I am moving and traveling around?
- What happens when I switch to Wi-Fi vs. 3G ?
- What happens if my Wi-Fi connection is down or intermittent?
- What happens when a mobile interruption occurs such as an incoming call or text message?
- What happens when the device is set to airplane mode?

SFDPOT

SFDPOT, heuristic by James Bach

T is for Time

Test how it's affected by time.

How I applied this to mobile testing:

- What happens if the time zone is switched?
- What happens when my location is switched?
- Does the device auto detect a change in time zones and reset the time (and possibly) the day and date?
- Does the app find time-related data based on my device time (such as reservation times available or movies show times)?

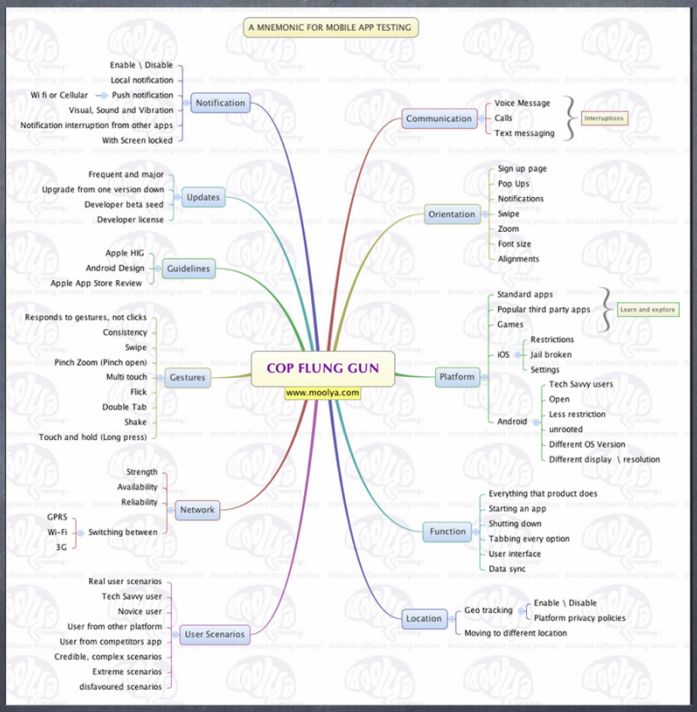
Moolya Testing blog post by Dhanasekar S

Another example -

A mind map and testing planned inspired from James Bach's SFDPOT (San Francisco Depot) and Jonathan Kohl's I SLICED UP FUN!

Source:

http://moolya.com/blog/?s=sfdpot



Applying Heuristics

First, learn about heuristics. Being knowledgeable about heuristics gives you tools to apply and encourages you to develop your skills.

Second, come to understand when a heuristic or method is helpful, when it should be used and when it should not be used.

Learn how to frame your own problem so you can apply the best heuristic to solve the problem.

- Adapted from “How to Solve It: Modern Heuristics” by Z. Michalewicz & D. Fogel

Developing Heuristics

- Look for ideas in testing that frequently get you results – perhaps some idea that often provokes the application you test and uncovers defects.
- “Time” is one idea I think of often. I work with timestamps and other date-sensitive fields (data replication testing, BI testing, etc.)
- Look for patterns, look for commonality.

Customize Heuristics

- Remember that you don't have to use someone else's heuristic in its entity – you can use part of a heuristic. For example, you might not find an applicable need for the "chronic" element in my regression heuristic. Feel free to drop part of a heuristic; keep what helps.
- Adapt someone else's heuristic to better help yourself.

Creating Mnemonics

- If a shortcut memory device helps you recall your ideas, then create a mnemonic.
- I found by thinking of three R's (recent, repaired and risky) and three C's (core, chronic and configuration), I was able to rapidly direct myself to testing that would uncover defects.
- Don't create a mnemonic if it doesn't help you!

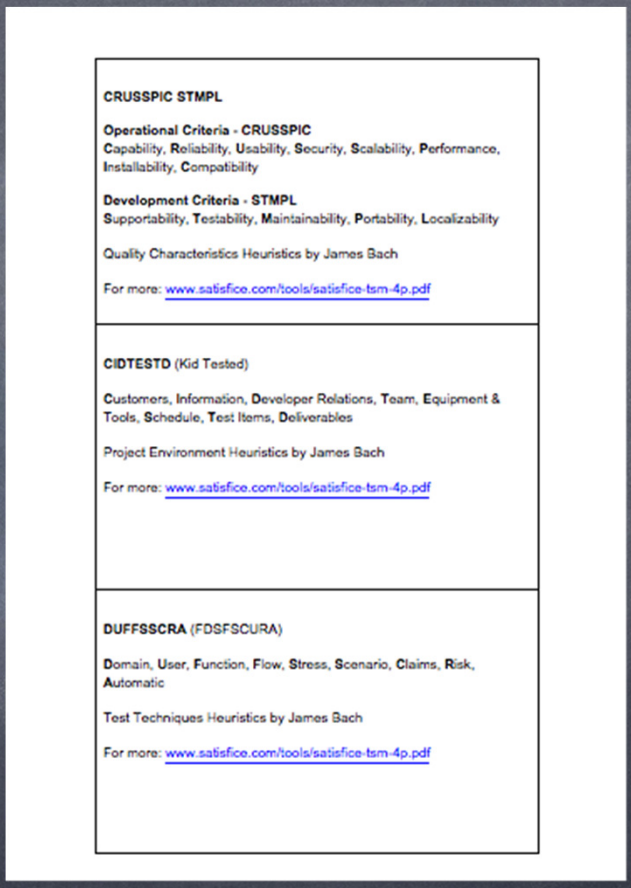
Karen Johnson a heuristic card deck

I collected all the testing heuristics I could find and formatted the heuristics to print on index cards.

I've made card decks for testers.

The purpose? To have a handy set of ideas easily within reach.

Download for free.



CRUSSPIC STMPL

Operational Criteria - CRUSSPIC
Capability, **R**eliability, **U**sability, **S**ecurity, **S**calability, **P**erformance, **I**nstallability, **C**ompatibility

Development Criteria - STMPL
Supportability, **T**estability, **M**aintainability, **P**ortability, **L**ocalizability

Quality Characteristics Heuristics by James Bach

For more: www.satisfice.com/tools/satisfice-tsm-4p.pdf

CIDTESTD (Kid Tested)

Customers, **I**nformation, **D**eveloper Relations, **T**eam, **E**quipment & Tools, **S**chedule, **T**est Items, **D**eliverables

Project Environment Heuristics by James Bach

For more: www.satisfice.com/tools/satisfice-tsm-4p.pdf

DUFFSSCRA (FDSFSCURA)

Domain, **U**ser, **F**unction, **F**low, **S**tress, **S**cenario, **C**laims, **R**isk, **A**utomatic

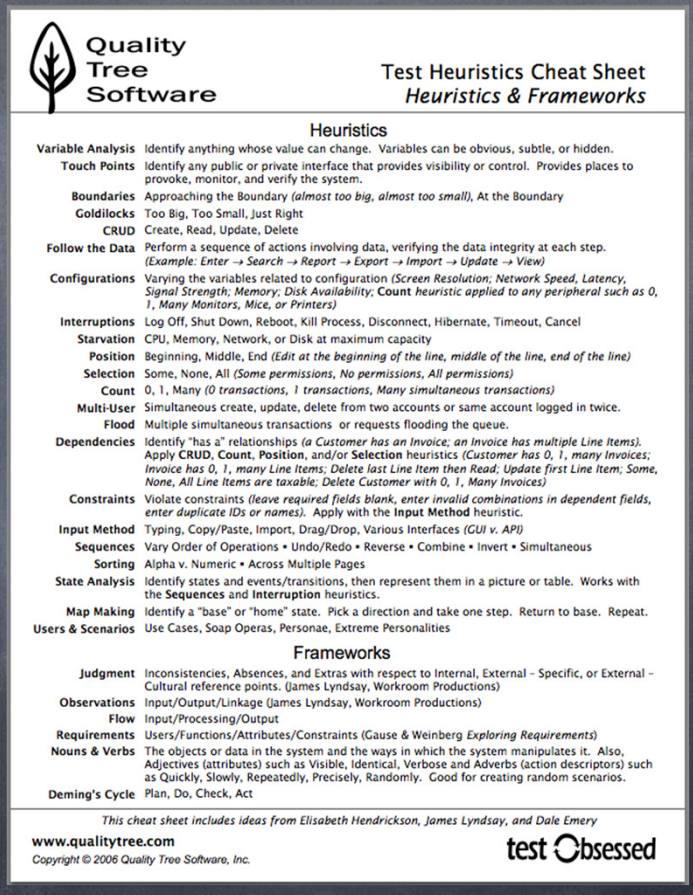
Test Techniques Heuristics by James Bach

For more: www.satisfice.com/tools/satisfice-tsm-4p.pdf

Elisabeth Hendrickson's "cheat sheet"

This two page cheat sheet has been around for several years.

You can download the cheat sheet as a PDF from her website.



Quality Tree Software

Test Heuristics Cheat Sheet
Heuristics & Frameworks

Heuristics

Variable Analysis	Identify anything whose value can change. Variables can be obvious, subtle, or hidden.
Touch Points	Identify any public or private interface that provides visibility or control. Provides places to provoke, monitor, and verify the system.
Boundaries	Approaching the Boundary *(almost too big, almost too small)*, At the Boundary
Goldilocks	Too Big, Too Small, Just Right
CRUD	Create, Read, Update, Delete
Follow the Data	Perform a sequence of actions involving data, verifying the data integrity at each step. *(Example: Enter → Search → Report → Export → Import → Update → View)*
Configurations	Varying the variables related to configuration *(Screen Resolution; Network Speed, Latency, Signal Strength; Memory; Disk Availability;* **Count** *heuristic applied to any peripheral such as 0, 1, Many Monitors, Mice, or Printers)*
Interruptions	Log Off, Shut Down, Reboot, Kill Process, Disconnect, Hibernate, Timeout, Cancel
Starvation	CPU, Memory, Network, or Disk at maximum capacity
Position	Beginning, Middle, End *(Edit at the beginning of the line, middle of the line, end of the line)*
Selection	Some, None, All *(Some permissions, No permissions, All permissions)*
Count	0, 1, Many *(0 transactions, 1 transactions, Many simultaneous transactions)*
Multi-User	Simultaneous create, update, delete from two accounts or same account logged in twice.
Flood	Multiple simultaneous transactions or requests flooding the queue.
Dependencies	Identify "has a" relationships *(a Customer has an Invoice; an Invoice has multiple Line Items).* Apply **CRUD**, **Count**, **Position**, and/or **Selection** heuristics *(Customer has 0, 1, many Invoices; Invoice has 0, 1, many Line Items; Delete last Line Item then Read; Update first Line Item; Some, None, All Line Items are taxable; Delete Customer with 0, 1, Many Invoices)*
Constraints	Violate constraints *(leave required fields blank, enter invalid combinations in dependent fields, enter duplicate IDs or names).* Apply with the **Input Method** heuristic.
Input Method	Typing, Copy/Paste, Import, Drag/Drop, Various Interfaces *(GUI v. API)*
Sequences	Vary Order of Operations • Undo/Redo • Reverse • Combine • Invert • Simultaneous
Sorting	Alpha v. Numeric • Across Multiple Pages
State Analysis	Identify states and events/transitions, then represent them in a picture or table. Works with the **Sequences** and **Interruption** heuristics.
Map Making	Identify a "base" or "home" state. Pick a direction and take one step. Return to base. Repeat.
Users & Scenarios	Use Cases, Soap Operas, Personae, Extreme Personalities

Frameworks

Judgment	Inconsistencies, Absences, and Extras with respect to Internal, External – Specific, or External – Cultural reference points. (James Lyndsay, Workroom Productions)
Observations	Input/Output/Linkage (James Lyndsay, Workroom Productions)
Flow	Input/Processing/Output
Requirements	Users/Functions/Attributes/Constraints (Gause & Weinberg *Exploring Requirements*)
Nouns & Verbs	The objects or data in the system and the ways in which the system manipulates it. Also, Adjectives (attributes) such as Visible, Identical, Verbose and Adverbs (action descriptors) such as Quickly, Slowly, Repeatedly, Precisely, Randomly. Good for creating random scenarios.
Deming's Cycle	Plan, Do, Check, Act

This cheat sheet includes ideas from Elisabeth Hendrickson, James Lyndsay, and Dale Emery

www.qualitytree.com

test Obsessed

References

- How to Solve It, by G. Polya
- How to Solve It: Modern Heuristics, by Z. Michalewicz & D. Fogel
- James Bach, blog post: The Essence of Heuristics http://www.satisfice.com/blog/archives/462
- Elisabeth Hendrickson. Cheat Sheet which includes some heuristicswww.testobsessed.com/wp-content/.../testheuristicscheatsheetv1.pdf
- Michael Bolton, blog post: Heuristics for Understanding Heuristicshttp://www.developsense.com/blog/2012/04/heuristics-for-understanding-heuristics/
- Karen N. Johnson, heuristic card deck www.karennicolejohnson.com/.../testing-mnemonics-as-a-card-deck/

References

- Jonathan Kohl, blog post: Learning Testing Heuristics http://www.kohl.ca/2006/learning-testing-heuristics/
- Adam Goucher, blog post: http://adam.goucher.ca/?p=200
- Michael D. Kelly, blog entries: http://michaeldkelly.com/blog/category/heuristics
- Lynn McKee, testing mnemonics www.qualityperspectives.ca/resources_mnemonics.html
- Daily Testing Tips, www.quicktestingtips.com/tips/category/heuristics/
- Moolya Testing http://moolya.com/index.html

Thank you for being here!

Karen Nicole Johnson

Email:
karen@karennjohnson.com

Site:

www.karennicolejohnson.com

or

www.karennnjohnson.com

- Questions?
- Comments?
- Can you use this?
- Does this help you?
- Please complete an eval form.