THE INTERNATIONAL MONTHLY FOR NEXT GENERATION TESTERS

Tea-time with Testers

DEC'13 + JANUARY'14 | YEAR 3 ISSUE XII

Jerry Weinberg

Tom Tireless tinkers with toys

Keith Klain

The Confidence Game - What is the mission of Testing?

Mike Lyles

Are you smarter than a Test Manager?

Jim Holmes

Test Automation - Working with locators

JeanAnn Harrison

Rants and Ramblings of Mobile Tester

Bernice Ruhland

Introducing SBTM To Your Team

T Ashok

Joy of Testing - Being Mindful & Mindless

Joel Montvelisky

Whoever you are, whatever you do, YOU ARE FIRED!

QA Intelligence
Testing & QA Management blog

Announcing the results of State of Testing Survey -2013

STATE *of* TESTING
2013

Grab your FREE copy from HERE

Sponsored by PractiTest

Best Speaker

TEA-TIME WITH TESTERS

First Indian testing magazine to reach 115 countries in the world !

Created and Published by:

Tea-time with Testers.
Hiranandani, Powai,
Mumbai -400076
Maharashtra, India.

Editorial and Advertising Enquiries:

Email: editor@teatimewithtesters.com
Pratik: (+91) 9819013139
Lalit: (+91) 8275562299

Opinions expressed or claims made by advertisers in this ezine do not necessarily reflect those of the editors of ***Tea-time with Testers***.

Editorial

Brand New Year with Brand New Ventures

Dear Readers

We are meeting bit late this time and I appreciate your patience. We felt overwhelmed after knowing how eagerly our readers wait for TTwT, each month and your love is something that keeps us going.

With the beginning of a new year we wanted to offer you something awesome and I am glad that we have figured out some more ways that will add to your reading experience with Tea-time with Testers.

So what are those awesome things? This year you'll get to read more interviews as we plan to do them bi-monthly rather than quarterly. We are also collaborating with some *smart* folks in industry who are working hard to offer you the best they have. Going forward, you will find some more surprises on the way and I am not going to disclose them from now ☺.

There is good news for sponsors too. We have added some more offerings to our professional services and that too at unbelievably low prices. Feel free to get in touch with our sales team (sales@teatimewithtesters.com) for more information.

By now, most of you must be done with New Year resolutions already but just in case if you haven't then please have a look on our 'State of Testing 2013' survey results. We are sure that you will enjoy the report and find it useful to plan your nest steps this year.

2013 was an interesting year and we look forward to make 2014 even more interesting!

Yours Sincerely,

Bhamare

- **Lalitkumar Bhamare**
editor@teatimewithtesters.com

QuickLook

Editorial

What's making News?

Tea & Testing with Jerry Weinberg

Speaking Tester's Mind

The Confidence Game - What is the mission of Testing? – 18

Are you Smarter than a Test Manager? - 20

In the School of Testing

Working with locators – 30

Rants and Ramblings of a Mobile Tester – 38

Introducing Session-Based Testing To Your Team – 42

Whoever you are, whatever you do, YOU ARE FIRED!!! - 49

T ' Talks

Joy of Testing - Being Mindful & Mindless – 47

Testing Puzzle – S.T.O.M. Contest

Family de Tea-time with Testers

Testing Puzzles
by Sebi

Crossword by

What's making News?

- find out the latest happenings in the technology world

mage: www.bigfoto.com

STeP-IN SUMMIT 2014
The 11th International Software Testing Conference

STeP-IN – India's largest Software Testing Community will hold **STeP-IN SUMMIT 2014 - the 11thInternational Software Testing Conference** on **June 26-27, 2014** at **Vivanta by Taj, M.G. Road, Bangalore** with single day pre-conference tutorial sessions at Bangalore, Hyderabad and Pune.

About STeP-IN SUMMIT 2014

Theme: **Testing NOW** – Today's tester is a unique being, driven, not by deadlines and mundane tasks, but by a relentless curiosity and a fascination with possibilities. Join us at STeP-IN SUMMIT 2014 – the 11th International Software Testing conference. Awaken the tester in you. This is where you want to be; this is NOW.

Why:"NASSCOM reported a growth of USD 3.5 billion for testing services in 2010 and a steady growth at 17 percent CAGR until the year 2020". Software Testing has become an integral and key component in all aspects of technology which drive the business world. Information in the Software testing arena is rapidly developing and this is a platform for Software testing professionals to interact and update themselves of the trends.

When:

Conference: June 26-27, 2014 @ BANGALORE

Pre-conference Tutorials: June18, 2014 @ PUNE

June 20, 2014 @ HYDERABAD and June 25, 2014 @ BANGALORE

Who is attending: CXOs, Vice Presidents, Directors, Business Heads; Test Management Professionals – Heads, Managers, Leads and Engineers; Development Professionals – Managers, Team Leads, Programmers; Quality / SEPG Management Professionals – Heads, Managers, QA Leads and Engineers; Customers and End-users of Products.

Who is speaking: Acknowledged gurus and Subject Matter Experts in the field of testing from leading organizations as well as startups.

Speakers at past STeP-IN events:

1. Kris Gopalakrishnan, CEO & MD, Infosys
2. Lee Copeland, Software Quality Engineering
3. Krishnakumar Natarajan, President and CEO, IT Services, MindTree Consulting
4. Dr. Ganesh Natarajan, Chairman, NASSCOM and Deputy Chairman & MD, Zensar Technologies
5. Shyamal Ghosh , Chairman, Data Security Council of India –a NASSCOM initiative
6. Michael Bolton, Founder, DevelopSense

Community Partners: Tea-time with Testers; QA Guild

Conference URL: http://stepinsummit.stepinforum.org/

About STeP-IN

STeP-IN, India's largest Software Testing Community forum in INDIA, was launched in October 2003. The forum aims at bringing together software test professionals, experts, academicians, & vendors to share techniques, methodologies, frameworks and experiences to perform, manage, and automate Software Testing.

STeP-IN has produced more than 20 conferences across different locations in India. STeP-IN SUMMIT has grown to become the largest Software Testing Conference in the Asia-Pac region.

More Details: www.stepinforum.org

For media, press and other information on the conference, interviews, one-on-ones with the Conference Speakers, please contact:

Nazreen Vakharia
Coordinator STeP-IN Forum
+91 99005 76905

nazreen@stepinforum.org
www.stepinforum.org

point blank

Are 'non-functional requirements' named such because that's what how your software ends up if you ignore them?

Adam Knight
Passionate Software Tester working with BigData Storage and Analysis systems in an Agile context.

It's fascinating & worrying seeing how many people reach my blog by searching for "how many test cases".

James Christie
Software testing consultant, ex-auditor

It is an extremely difficult ask to expect someone to believe you take testing seriously, when you use "QA" as a verb.

Keith Klain
Head of Barclays Test Service

There is a difference in receiving something that is broken and finding a way to break it.

Jari Laakso
A Finnish context-driven software testing enthusiast.

Metrics can totally skew our understanding, depending on what we choose to count - and what we leave out

Fiona Charles
Software test consultant, teacher, writer, speaker, iconoclast

Tom Tireless tinkers with toys

We know that most people, most of the time, feel they have some sort of problem. By our broad definition of "problem," these people must be correct, for a problem is a difference between someone's desires and the way things seem to them.

Knowing you have a problem is a matter of feeling. If you feel you have a problem, you do have a problem. Knowing what that problem is—well, that could be another matter.

To be sure, most people with problems also think they know what these problems are. In this, however, they are usually wrong.

A cleverly paradoxical example of such a mistaken impression is the belief that "problem solving" is a big problem. Many people have told us, "My major problem is that I'm not a very good problem solver." Pish-posh! More often than not, solving, or resolving, a problem is a rather trivial exercise—once we know what the problem is. Possibly the reason schools turn out such poor problem solvers is that students are never given the chance to find out what the problem is—the problem is whatever the teacher says it is. You'd better believe it!

Most of us have had schooling—too much of it. We've developed an instinct that makes us seize upon the first statement that looks like a "problem". Then we "solve" it as fast as we can, for everybody knows that, on exams, speed counts. And so does concentration.

As a result of all this exam-training, we've learned problem habits that are rather difficult to overcome when we're not in school, taking exams.

And don't mistake what we're saying. The approach of grabbing the first problem statement, digging in fast, and sticking with it to the bitter end is precisely what you want—when you're trapped in a school system and trying to make the best of a bad business. And even in a few other situations. In fact, the Brontosaurus Tower problem might have been resolved in a trice, had someone leapt to the conclusion that "the elevators are too slow and need to be fixed." The "two foot blind leap approach" works just often enough to keep it alive. If it never worked, people would eventually stop using it—when they'd been out of school long enough.

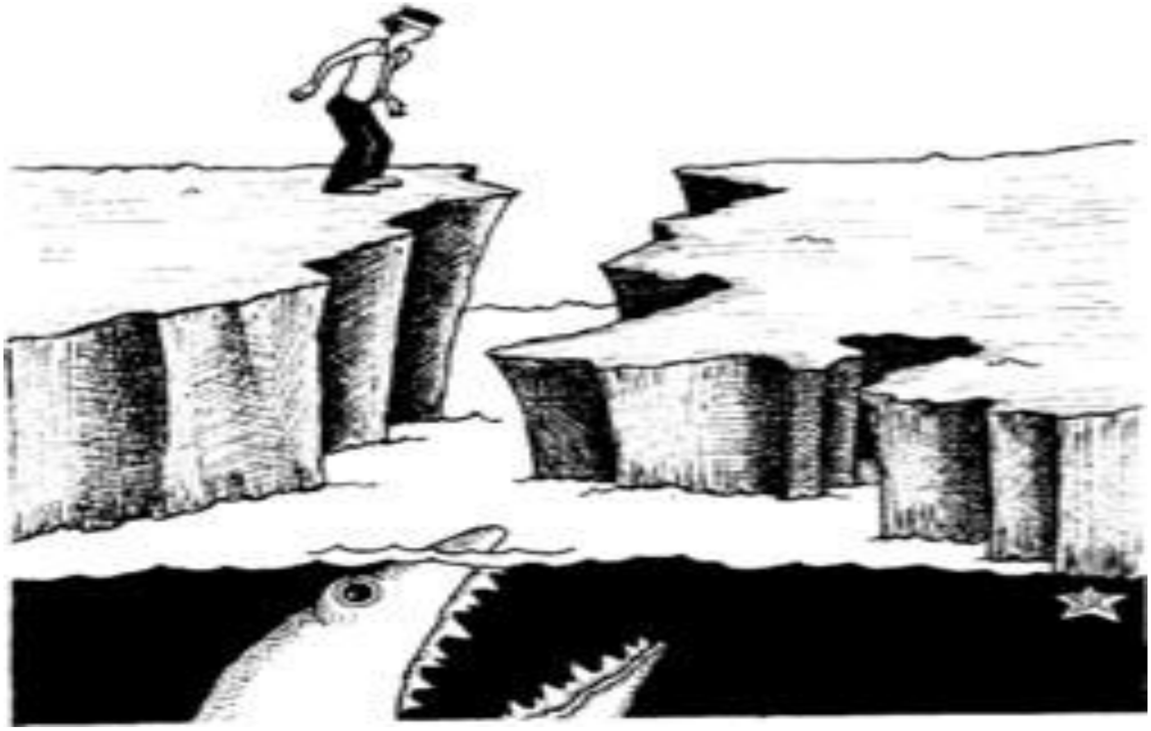


Figure 1. Sometimes, the two-foot blind leap approach works.

Another reason the "two-foot blind leap approach" persists is that "problem solving" can be such fun. Once we get going on a smashing problem, only a pervert would want to get in our way. We all know the type: one of those spoilsports who actually gave up smoking when the Surgeon General's report was issued; who thinks all others should follow his lead; and who never loses an opportunity to preach to them about it.

Why, even if what we're solving isn't the "real" problem, it becomes the real problem to us because we want to solve it—the more heroically the better. In other words, leave us alone! Who gave Don and Jerry the moral right to interfere with other people's problem-solving fun?

It's a good question, and one we'd better face, because Don and Jerry are two of the world's foremost problem-solving spoilsports. Our moral right comes from the admonition,

DO UNTO OTHERS AS OTHERS HAVE DONE UNTO YOU.

Each of us, and many of you, have had our fun spoiled by some eager young problem solver disturbing our peaceful equilibrium. That's what gives us the right to spoil some of the problem solver's fun.

What do we mean by disturbing our peaceful equilibrium? A good example can be drawn from the computer field. When computers first began to be abundant, people did not exactly beat a path to their inventor's door. To an alarming extent, computers were pushed on an unwilling, or at least cautious, public—promoted by an enthusiastic corps of problem solvers specializing in the application of computers to just about anything.

They were young, these problem solvers, with all the impetuosity of youth. Their slogans reflected their youth:

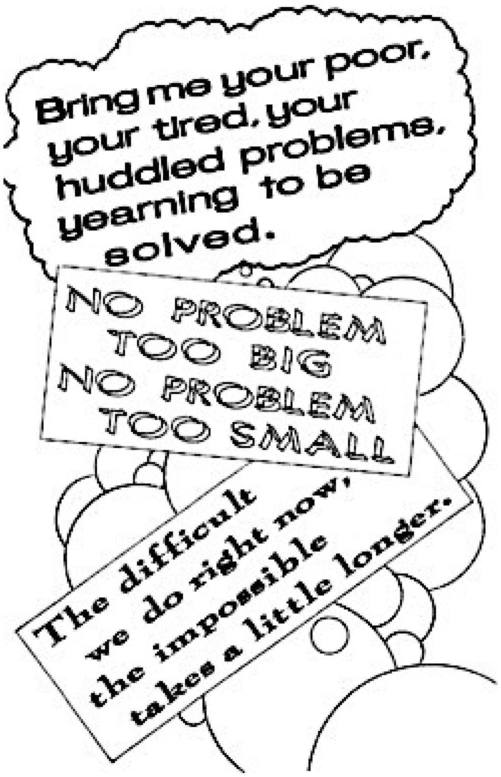

Bring me your poor, your tired, your huddled problems, yearning to be solved.
NO PROBLEM TOO BIG NO PROBLEM TOO SMALL
The difficult we do right now, the impossible takes a little longer.

Figure 2. We problem solvers had our youthful slogans.

Computers have that effect on youth—or, rather, youth has that effect on computers.

Why, even with several decades of accumulated wisdom to say it isn't so, each new school of hatchlings spilling out of their first programming course is ready to move the earth—given only enough computer time and a terminal to sit at. After all, what has youth ever cared for musty, dusty accumulated wisdom?

And, after all, aren't they right? If nothing else, these young computniks will acquire one valuable lesson from their unrelenting quest for problems to fit their solution—"solution probleming," we call it. As they quest, so shall they learn. Mostly, they'll learn about problem definition.

They will learn how hard it is for people to state their problems clearly enough to satisfy the precise demands of the computer, for whom the most trivial steps must be spelled out in pettifogging detail.

At first, these hatchlings will decide that people are no darned good at communicating—and, at times, this pessimistic assessment will be correct. But more often than not, communication won't be the source of the difficulty. We can't communicate what we don't know—or don't want to know.

But enough of moralizing. What about an example? Once upon a time, the enthusiastic young programmer, Tom Tireless, happened upon a toy factory. His effervescence easily floated him past the outer defensive rings surrounding the executive suite. Soon, he found himself seated in an elegant conference room with three vice-presidents. In a few minutes, Tom had them greedily counting the blessings about to be showered upon them by his computer.

After some preliminary education of these novices about the powers of his magical device, Tom, our solution-problemer, asked the vice-presidents if they had any problems lying about that might prove suitable. Yes, as it happened, they did have a most pressing problem. It seems that Tanglelang Toys (TT) had three factories—this one on the Pacific Coast, another one on the Atlantic Coast, and a third on the coast of the Missouri River, at Kansas City. From these factories, they shipped toys to about 50 wholesalers, scattered throughout the country more or less as shown on this map:

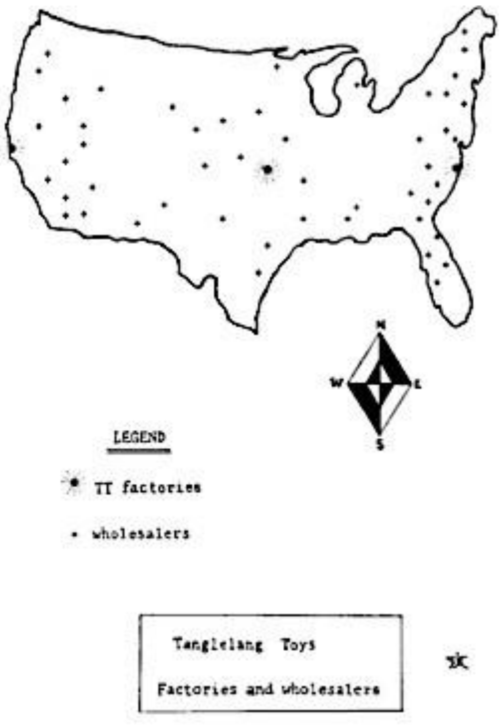


Figure 3. The distribution of TT factories and wholesalers.

Naturally, the executives explained, shipping costs money, thereby adding to the effective cost of each Tanglelang Toy. Moreover, because different wholesalers varied in their shipping distance from the different factories, all shipping costs were different. By this time in the explanation, Tom Tireless was getting restless. He wasn't used to sitting for very long, except at a computer terminal. Certainly not sitting and listening.

Tom had long since recognized what their problem was going to be—a classical problem in operations research, one that could be handled niftily on his computer. While they dragged out their explanation, Tom tuned out He was solving the following problem:

Given a set of orders from their wholesalers, how should TT allocate these orders among the three factories so as to minimize the total cost of doing so—manufacturing plus shipping costs.

By the time the vice-presidents finished explaining that this was indeed their problem, Tom was ready to request the information the computer would need: the total set of orders, the cost of making each toy at each factory, and the cost of shipping each toy from each factory to each wholesaler.

It took the TT executives some time to obtain this information for Tom, but two weeks later it lay neatly assembled on his desk back at the computing center.

Tom spent some time checking the figures this way and that. He began to notice a disturbing pattern. When he finished, he telephoned the executives for an appointment.

"I'm sorry to report," Tom announced, "that I have discovered something curious about your problem. If your figures are correct, then it is possible, for example, to make a teddy bear in this factory and ship it to the Kansas City factory for less than the cost of making it in Kansas City! Their cost is $3.95 and yours is $3.07. If you add $.23 for shipping, that makes $3.30, or 65 cents less than their manufacturing cost alone."

Tom laid both hands on the conference table for emphasis. The three vice-presidents sighed and exchanged glances. "Yes," replied the most senior of the three, "we know that."

"And do you also know that the same hold true for the Atlantic Coast plant?" He paused to let his words sink in. "That you can ship the teddy bear there for $3.38—and they can't make it for less than $4.24?"

"Yes, we're aware of that too. Just what are you driving at, young man?"

"Please, one more question. Are you further aware that this same pattern holds true for every one of the 374 toys in your line?"

"But of course we're aware of that. This factory is the most modern in the world—far more efficient than the other two, with lower labor costs, to boot. That's why we built it here."

Tom was perplexed at their density. "But don't you see? You don't need a computer to show you how to reduce your cost to the lowest possible level." It was painful for Tom to admit this, but he continued. "All you have to do is board up the other two plants! Make all your orders here, and ship them from here! Why, even if you didn't ship directly to the wholesalers, but first sent to the shipping docks of your other factories, it would have to be cheaper than the way you're doing it now."

"That's true. But we can't accept that solution."

"What? A solution is a solution. Why can't you accept it?"

"Because the President of TT lives near our Atlantic Plant. And the Chairman of the Board lives in Kansas City. Why, they wouldn't move to the Pacific Coast for anything."

"They certainly wouldn't," the others agreed in unison.

"But then your problem is not one of minimizing costs, but of making your President and Chairman happy?"

Our solution-problemer was exasperated. "Then why did you give the problem to me?"

After pondering that question for about thirty seconds, the senior vice-president said, "You said your computer could solve any problem. I suppose we wanted to be convinced that the computer could help us. We've known all this ever since this plant was built, but we haven't been able to convince our two most senior officers we were right. Yes, I suppose we thought that if your computer told them, they might believe it—even though they wouldn't believe us. But now that we think upon it more clearly, that doesn't seem valid."

Tom was almost crushed, but couldn't quite let go. "Why not? I can run these figures through my linear programming package, and the computer will give you a terrific report—even printed with lots of mathematical symbols that can't fail to convince your executives. Just give me a chance."

The executive continued, hardly noticing the interruption. "No, they simply aren't going to move, regardless of the cost to the company. They can afford it. We three, on the other hand, would like to see our business run more efficiently—we haven't yet made our fortunes."

Figure 4. Tanglelang Toys Teddy Bear

And thus Tom Tireless learned lesson number one in problem definition for those who would presume to solve problems for others:

IN SPITE OF APPEARANCES, PEOPLE SELDOM KNOW WHAT THEY WANT UNTIL YOU GIVE THEM WHAT THEY ASK FOR.

Back To Index

Biography

Gerald Marvin (Jerry) Weinberg is an American computer scientist, author and teacher of the psychology and anthropology of computer software development.

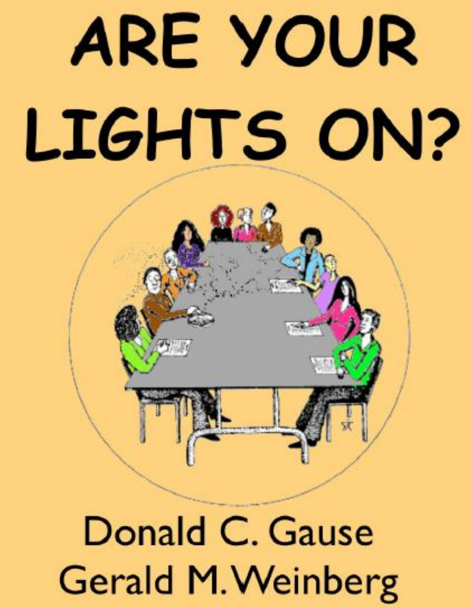
For more than 50 years, he has worked on transforming software organizations. He is author or co-author of many articles and books, including The Psychology of Computer Programming. His books cover all phases of the software life-cycle. They include Exploring Requirements, Rethinking Systems Analysis and Design, The Handbook of Walkthroughs, Design.

In 1993 he was the Winner of the **J.-D. Warnier Prize for Excellence** in Information Sciences, the 2000 Winner of **The Stevens Award** for Contributions to Software Engineering, and the 2010 **Software Test Professionals first annual Luminary Award.**

To know more about Gerald and his work, please visit his Official Website here .

Gerald can be reached at hardpretzel@earthlink.net or on twitter @JerryWeinberg

ARE YOUR LIGHTS ON? is one of the famous books Jerry has written together with Donald C. Gause.



TTWT Rating: ★★★★★

ARE YOUR LIGHTS ON? has received great feedback from readers and we strongly recommend you to read it if you want to get 'problem solving' right, of course along with many other interesting insights that this book offers.

Its sample can be read online here.

To know more about Jerry's writing on software please click here .

Speaking Tester's Mind

- straight from the author's desk

The Confidence Game- What is the mission of Testing?

by Keith Klain

Doubt is not a pleasant condition, but certainty is absurd. - Voltaire

Maybe it's due to an extension of my tendency towards skepticism to myself, but I get really uncomfortable telling anyone that something is certain. That is especially true when it comes to software and interpreting the results of testing. There are just too many variables that impact the control and validity of the output, and that's just limited to what we can know – let alone the things we don't know! The great "unknown unknowns" loom in the shadows, waiting to rear their head and question our approach and as well – shake our confidence.

By definition, confidence is the quality or state of being certain. It's knowing that something can be proved true, and is a by-product of actions taken in the process of acquiring that proof. Christopher Chabris and Daniel Simons created a famous experiment in studying inattentional blindness. In their book The Invisible Gorilla, they posit that we should be very ***unsure*** of what we are certain we know, and that our confidence or intuition can often mislead us. The idea of questioning the origins of our confidence is also echoed in Blink: The Power of Thinking Without Thinking by Malcolm Gladwell, and Thinking, Fast and Slow by Daniel Kahneman.

So what does that have to do with the mission of testing? It is extremely important that testers understand and adhere to their mission, as to replace it (either willfully or unintentionally) would be directly fogging the headlights on your project. So should the mission of testing be to give confidence? I don't believe it should. I would agree with my friend Michael Bolton, that making "confidence" your mission in testing is akin to goal displacement, or substituting objectives with those that suit your means as opposed to the end.

I believe the mission of testing is gaining information; but here are some better examples for your reference:

- Testing is a process of technical investigation, intended to reveal quality-related information about a product (Cem Kaner)
- Testing is questioning the product in order to evaluate it (James Bach)
- Gathering information with the intention of informing a decision (Gerald Weinberg)

So what is the problem with making confidence the mission of testing? Shouldn't we want to have confidence in our products? Isn't it a good thing to have confidence in our testing? Of course we want confidence in our products and testing, but if you make gaining that confidence your mission, in my opinion, you are **intentionally** adding confusion to the decision-making process. Aside from trying to hit the bulls-eye on the wrong target, testing for confidence is a slippery slope to ill-informed decisions, misuse of metrics, and a ready candidate for confirmation bias.

Testers should be constantly vigilant against all forms of bias, but especially confirmation bias. Making confidence your mission guarantees you will be seeking information to give your stakeholders certainty – instead of information that should give them pause for thought. Every tester has at times been subject to the "Curse of Cassandra", or giving a valid warning that is not heeded. But nothing will put you permanently in that place quicker than having things go wrong after you've not only provided information to stakeholders – but have made a value judgment on their behalf!

Some may view me as overly skeptical. That's fine. But I would rather err on the side of caution (and humility) when seeking information for my stakeholders. It's up to them to decide what to do with what I give them – objectively finding it is hard enough without attempting to gain credibility through inappropriate means. So when someone tells me someone has asked them to give them confidence through testing, my simple advice to them would be this: stick to the mission.

Keith Klain is the co-head of the Barclays Test Service, which provides quality assurance and software testing activities across all asset classes and businesses for 1700+ employee and vendor staff in Barclays Bank. With nearly 20 years of multinational experience managing enterprise-wide testing programs, Keith has built and managed global test teams for financial services and IT consulting firms in the US, UK, and Asia Pacific.

Keith is a current member of the board of directors for the Association for Software Testing and was the recipient of the 2013 Software Test Professionals Luminary award.

Visit his blog at www.qualityremarks.com

Back To Index

Are You Smarter than A Test Manager?

- part 2

- by Mike Lyles

Welcome back to the second installment of my two part series on test management. If you read last month's article, we covered the job description of test managers, the 'true' (and humorous) job description of test managers, and we asked four questions to multiple testing professionals in the field. We found that while everyone has different views and positions on many aspects of testing, we have more in common than we realize.

This month, we move beyond the team of experts and we examine a survey that I conducted over a year ago. I encourage you to join me as we recap those responses and review the trends that were found in this process.

The Survey

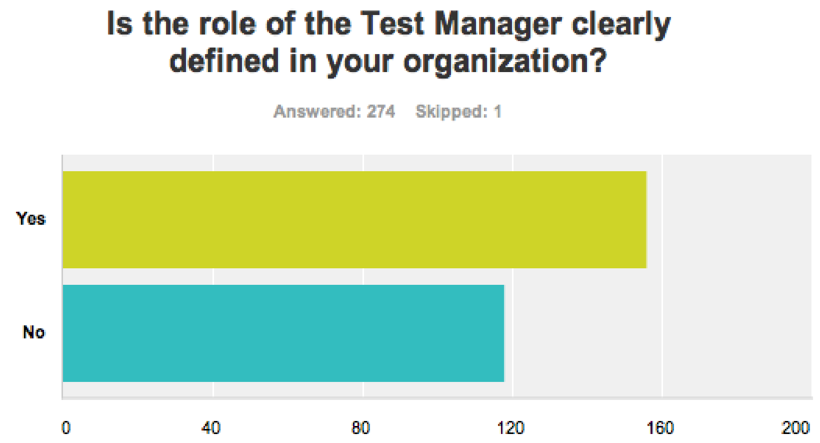
I combined thought provoking responses I gathered from interviewing the testing experts with additional questions on the test management process and created a survey on test management, which was taken by 275 people across 36 countries.

Of the many questions around how managers spend their time, one that stood out to me was the time spent in 'process improvement' as noted below:

Activity	**0%**	**0%-25%**	**26%-50%**	**51%-75%**	**76%-100%**
Process Improvement of QA	7.7% (21)	69.3% (190)	15.7% (43)	6.6% (18)	0.7% (2)

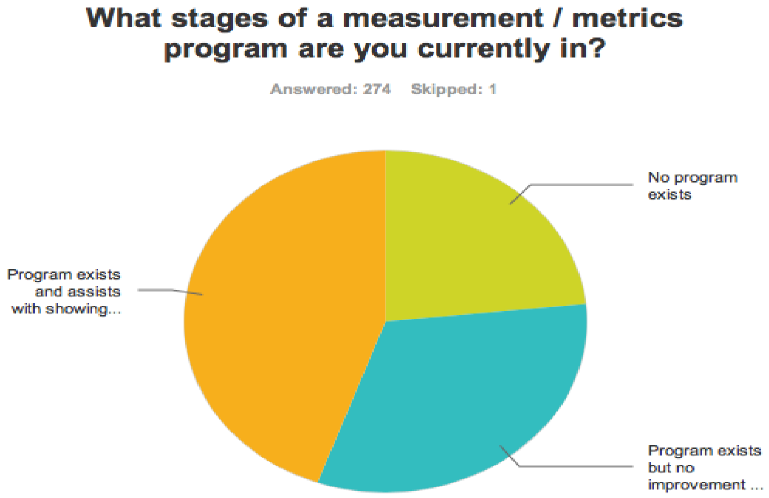
It was surprising to see that 7.7% of those surveyed stated that they spent 0% of their time improving processes in their organizations. When you combine that with the 69.3% that stated they spent 0-25% of their time in process improvement, we have 77% of those surveyed spending less than 25% of their time around process improvement.

With my inputs from the experts, I asked those surveyed if they felt the role of test manager was clearly defined and received the following results below. To me, it was eye opening to see that 43.07% of those surveyed felt the role was not clearly defined in their organizations:



Answer Choices	Responses	
Yes	56.93%	156
No	43.07%	118
Total		274

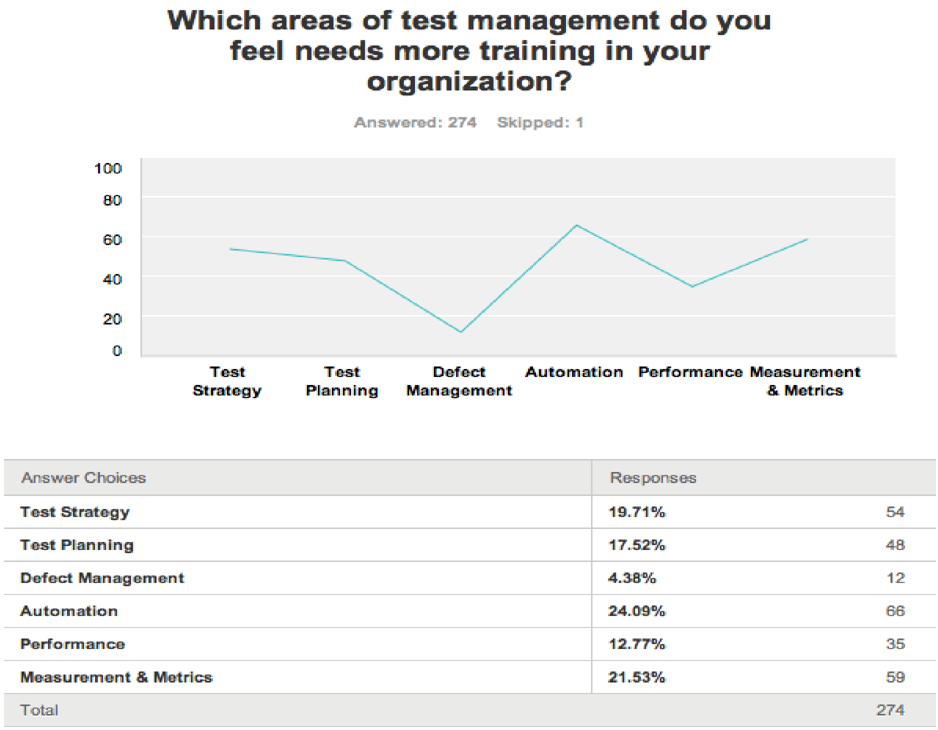
I then asked about the way that measurement and metrics were used in their organizations.



Answer Choices	Responses	
No program exists	23.36%	64
Program exists but no improvement to the overall process	32.12%	88
Program exists and assists with showing the value of the organization & quality of deliverables	44.53%	122
Total		274

It seems surprising that 23% of those surveyed have no measurement/metrics program in place at all. And another 32% felt that even though a program exists, there is no improvement due to having one. We are surely doing a disservice to our organizations if we are not leveraging a way to report the quality of the product being delivered in some form. We can debate whether one metric verses another is reliable, valuable, or even justifiable. However, there must be a way to explain what has been executed and the corresponding results.

Another question I asked was where people felt training was most needed within test management. The following were the results:

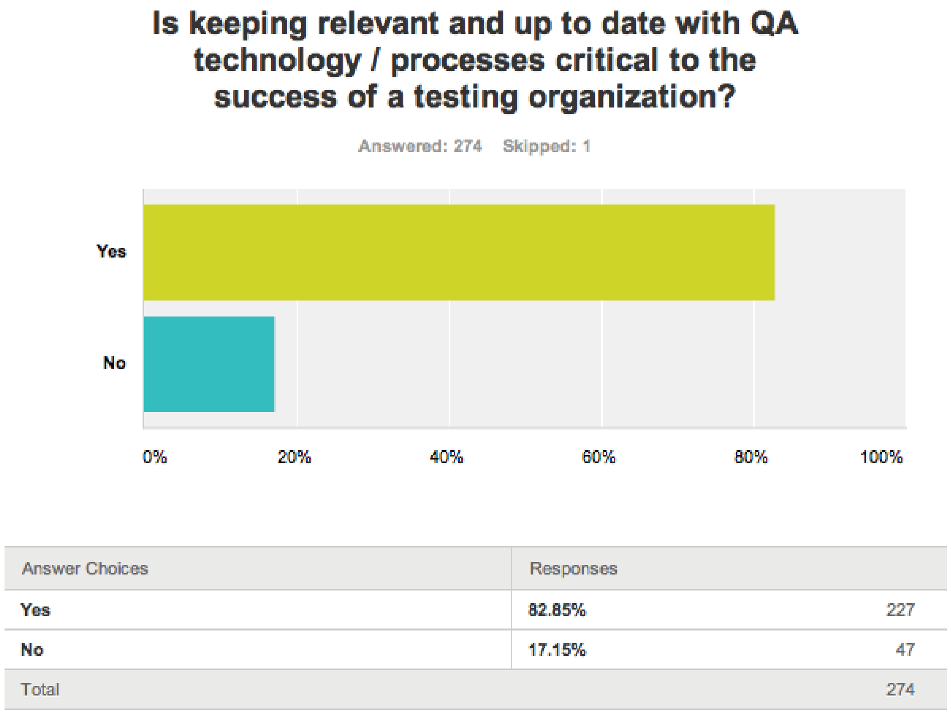


Which areas of test management do you feel needs more training in your organization?

Answered: 274 Skipped: 1

Answer Choices	Responses	
Test Strategy	19.71%	54
Test Planning	17.52%	48
Defect Management	4.38%	12
Automation	24.09%	66
Performance	12.77%	35
Measurement & Metrics	21.53%	59
Total		274

The next question was around keeping relevant with latest technology and processes around QA:



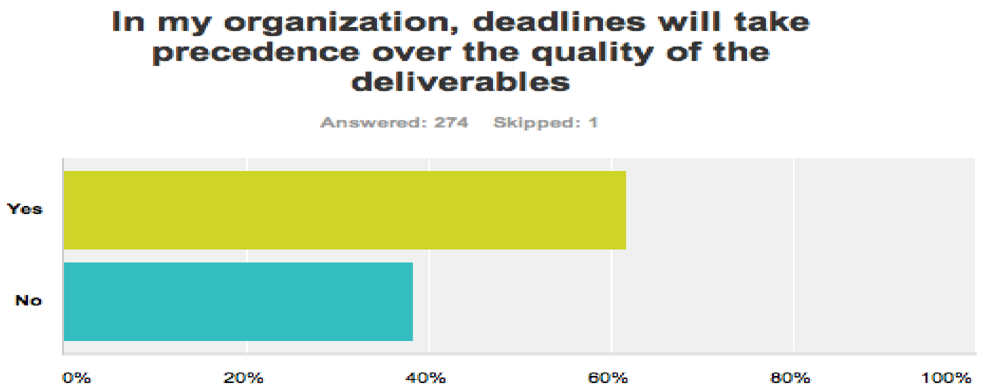
Is keeping relevant and up to date with QA technology / processes critical to the success of a testing organization?

Answered: 274 Skipped: 1

Answer Choices	Responses	
Yes	82.85%	227
No	17.15%	47
Total		274

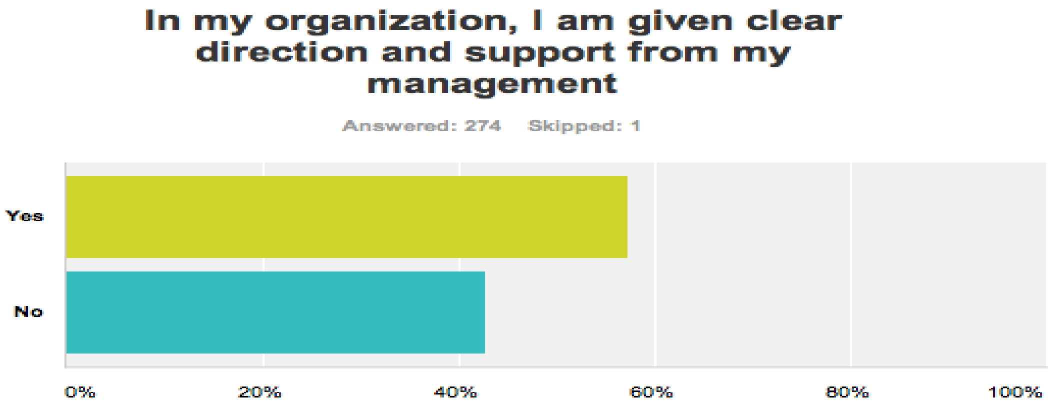
This was good to see, however, these are the same survey takers where 7.7% stated they spent 0% of their time on improving their processes, and 69.3% stated they spent 0-25% of their time improving their processes.

So how do deadlines affect the quality of your organization's deliverables? When I asked this in the survey, I received the following response – showing 61.7% of those surveyed stating that deadlines take precedence over quality.



To truly be effective as an organization, we must obtain buy in from all of the stakeholders, especially those accountable for delivery to production, and never let quality be sacrificed in order to meet a deadline.

With the next two questions, I really enjoyed the results, and wanted to share how we, as leaders, think. When asked if they feel they are given clear direction in their role – 42.7% stated they did NOT.



The next question asked was whether or not we give clear direction to our team – the response was an overwhelming 93.8% to "YES".

Now, I must question each of us with this…If you are not given clear direction in your job, how are you able to pass the clear direction on to your teams? And how are you sure that the direction is accurate? This should raise flags in all of our organizations if we are in this position and we feel we are not getting clear direction. Communication is key. Begin leading up, across, and down, and you will see a better vision for your organization.

This leads me to my section of the survey where I gave folks the opportunity to answer three questions in their own words. I received hundreds of responses and I want to share with you some of the highlights of the responses I received for those three questions.

QUESTION 1: Why Did You Become a Test Manager?

"**"What other job can you be loved and hated at the same time?"**This was one of my favorite quotes. I think we all agree that it's a tough role, and if you're truly doing an awesome job, you're probably not making a lot of friends, especially with those whom you are exposing the issues with their product.

"**I was given the option to take a full training in Test Management as a dev manager, and during my training, it became apparent to me that I was meant for testing**" I love this one because it's similar to how I got my start. I was a development manager and was given the chance to attend a week long testing course many years ago, and it simply opened my eyes that without a strong testing organization, the development team can never be successful and the product can never have a healthy chance of being of high quality.

"**Testing is my passion.**" One of the most common responses was regarding a passion for testing, and quality. I believe this is very true. With many occupations, you can do the job without truly loving it. But I would be willing to bet that almost every testing professional DECIDED to be in testing. They didn't just choose it to pay the bills or have a job. There is something to be said about this fact. We need passionate people who truly care about the quality of the product to deliver.

"**I like breaking things**" I have bad news to all testing professionals who believe this way. We do not break anything. It was given to use broken. We simply found it.

"**I know the Test field specifics perfectly well, understand the testing value in the business, I enjoy working with people and like to stand up processes**" As mentioned earlier, I have found, in my research and interviews, that testing is a highly social profession. People have to enjoy working with others, understanding the solutions and the product, and communicating throughout the organization. If you're not a social person, your success in the testing profession is probably not very high.

"**I wasn't satisfied with the influencing power I had over decisions as a tester**" I can understand why someone would believe this, but to everyone reading this, I must warn you – if you feel that moving into a higher position will give you more influence, you will most always be sadly surprised. We can lead up as much as we can downward and across. Do not depend on a promotion to give you the influence you need to be successful. Be influential regardless.

"**To seek for authority and be a decision maker in testing. To fight for justice in quality in terms of delivery**" The way this person wrote this comment made me smile. Because all I could see in my mind was someone in a Superman suit and a big "Q" on their chest instead of an "S". Fighting for Justice!

"**I would say testing by itself is a project management activity and requires greater vigor, communication skills and an eye for detail throughout the project life cycle**" I have to agree with this. We have our own little "project within a project" with our testing activities.

“**Guide and shape how testing was done, be a ‘louder’ voice for good testing**” Now THIS is a great reason to become a test manager! This is one place we CAN be a guide, to the testing team, and across the management peers in other groups.

“**Easiest bit of SDLC**” Ok, this has to have been the one development person who took my survey. I don’t think anyone in the testing profession would agree that it’s the easiest bit of the software development life cycle.

QUESTION 2: What Things Would You Do Differently If You Could Decide How Your Day Would Look?

“Nothing. When I see a better way to do things I don't wait around. No meetings, no attempt at consensus, I just get it done. Sometimes slowly and gradually, sometimes not. People who have a list of things they would do differently are not managers, they are observers.”I understand the passion and drive that this person had in writing this comment, but even the best leaders in the world have a game plan. And no matter how successful we are, we must always have a list of things we could do better or differently to improve.

“**I would spend time trying to open the minds of my peers who have resigned themselves to continuing to do exactly what they know right now, with no desire to learn new things.**”I’m betting that most everyone reading this article can relate to this statement. Stephen Covey had a quote “Nothing fails like success”. He talked about how that things we do today to be productive and successful will more than likely not be good enough for success as the world changes and things change. If you are struggling with this in your organization, you must fight to show the value of continuous improvement and growth across the testing organization, or the team will be replaced with those who will.

“**Align QA metrics with business objectives.**”Great suggestion. Challenge your teams to stop focusing on the number of test cases executed per hour, the % pass vs. % fail, and more on aligning with the company objectives and how we show the focus of quality to those objectives.

“**Ensure that more test hardware is available earlier in projects.**” There is a reason why cloud technology and virtualization are becoming more popular every day. The testing team is almost always the lowest priority in obtaining environments.

“**Implement TCoE model for a cost effective testing team**” A testing center of excellence is not about cost effectiveness, although that may be a result of having a TCoE. But a TCoE is more about process improvement, standardization, methodology, and more.

“**I would be demanding a monetary value on all requirements**” I really liked this concept. Imagine having a monetary value on all requirements, and the support this would provide in a risk-based testing approach, and determining which requirements need the most support in validation. Could your organization do this?

QUESTION 3: What Topics within Your Organization Do You Feel There is a Lack of Alignment / Agreement?

"**Lack of agreement on who owns the decision to move to production**" My belief, and I am quoting Michael Bolton in a discussion I had with him once, is that we, the testing organization, are investigative reporters. We report the news – we don't solve it or change it – we simply report it. Obviously, if you know that a release will greatly impact your organization, you should strongly suggest that it not be moved into production. But to own that decision…that's a lively debate in many organizations.

"**Manual testing - management thinks that manual testing is a burden and test automation is the only efficient way and will reduce costs**" Have you ever heard anyone use the term "manual development"? Then why are we using it in testing? All testing is manual. Automation is a tool to help the process.

"**Importance of getting certified**" Anyone seen this debate lately? How does your organization feel about it? I will tell you that in the early years of testing, this was a necessity. Is not now? That's something for you and your organization to decide. And I plan to have many forums, and likely webinars, on this very topic in the coming year.

"**Automation reduces the cost of testing to near zero**" Automation is not about speed or cost. It is always about efficiency, repeatability, and consistency. If your organization is hanging their hat on automation to solve the headcount on the team, or to reduce spending on testing efforts, then you may be in for a big surprise. Automation is needed, and it truly helps teams to be more efficient. This is coming from an automation manager, so I know this to be true. But educate yourself on the value of automation, choose your automation efforts wisely, and make sure your stakeholders understand this as well.

"**The ever complicated question of what is 'good enough' quality**" If your first thought, when reading this quote, was 'how can the testing team improve', then you're already failing. "Good enough" is not only focused on the work of the testing organization. It needs to be shifted left into development, and the quality of the code needs to be addressed before we even move to test. Ensuring your organization has a good code quality metrics program in place can assist here.

"Quality - a lot of talk about the value of testing early and the reality of not consistently getting the test team involved early enough in planning or requirements huddles" We've all said it. You have. I have. Everyone has. But my suggestion is to be careful what you ask for. If you continually say "if testing was involved earlier in the process, we wouldn't have these issues", then you might get a response that you are unprepared and unable to staff in the early stages of the project. Think less about "shift left" and more about "starting right".

As you can see, I obtained a lot of awesome feedback from many people in my search for the great solution for test managers. And we're still learning. Just look at how we did things years ago verses today. The world continues to evolve and grow faster than most can keep up. We deal with social networking, digital consumerism, pervasive computing – and the need for anywhere, anytime, any device systems. Big data analytics is around us everywhere, analyzing our buying habits, and how we live our lives, and then making suggestions to us, as consumers.

How do you use all of this information? How do you socialize it within your team?

- I suggest that you find your passion. Is it truly testing? If so, then keep moving, and find others who share your passion, and learn together.
- I suggest that you choose your battles. Ask yourself if the things you are fighting for today truly matter, or if letting them go and moving on may be more beneficial to you and others.
- Remember that the Drive-Thru is not always faster. I say this from experience, sitting in my car and watching people go into the fast food restaurant, buy their food, and then leave while I'm still in line. Applying this to your daily life – don't settle for what may seem like the "best and fastest solution". Sometimes the best solution is not always what is perceived to be the fastest.

As testers we need to stand up, be proactive, and respond, not wait until our turn and be reactive. That will not get us the respect we deserve.

If you want to make a change in your organization you have to STAND UP. How do you stand up? By getting CONVICTION. How do you get conviction? By obtaining KNOWLEDGE. How do you get knowledge? Try some of these:

- Watch the organizations (AST, ISST, ISTQB, QAI, etc.)- whether you agree with them or not
- Join LinkedIn groups
- Attend conferences
- Follow the Twitter community (find me there **@mikelyles**)
- Talk to the testing experts / leaders
- Read BLOGS – contribute and learn
- See (**http://www.mikewlyles.com/testapedia**) for links

It has been a pleasure to share this information with each of you. Please reach out to me anytime. You can find all of my social networks at **http://about.me/mikelyles** or go to my website at **http://www.mikewlyles.com**

Mike Lyles is a Sr. QA Manager with 20+ years of IT experience, working in various roles over the years. His current role is over Performance and Automation testing for all business communities within his organization. Mike enjoys teaching others in the testing profession. He has spoken at multiple conferences on test management topics, and has written multiple published articles.

You can learn more about Mike at http://about.me/mikelyles or on his website at www.MikeWLyles.com.

Back To Index

There was a time when people did not have compass to find right direction. The only guide they had was that guiding star up in the sky.

Do you think that you are also stuck somewhere with technical issues? Do you need help in decision making or want guidance?

Well, the wait is now over . Introducing….

"The Guiding Star"

The panel of our experts is now here to help you.
Send us your questions around software testing and our Guiding Stars will help you out.

E-mail your question on –

theguidingstar@teatimewithtesters.com

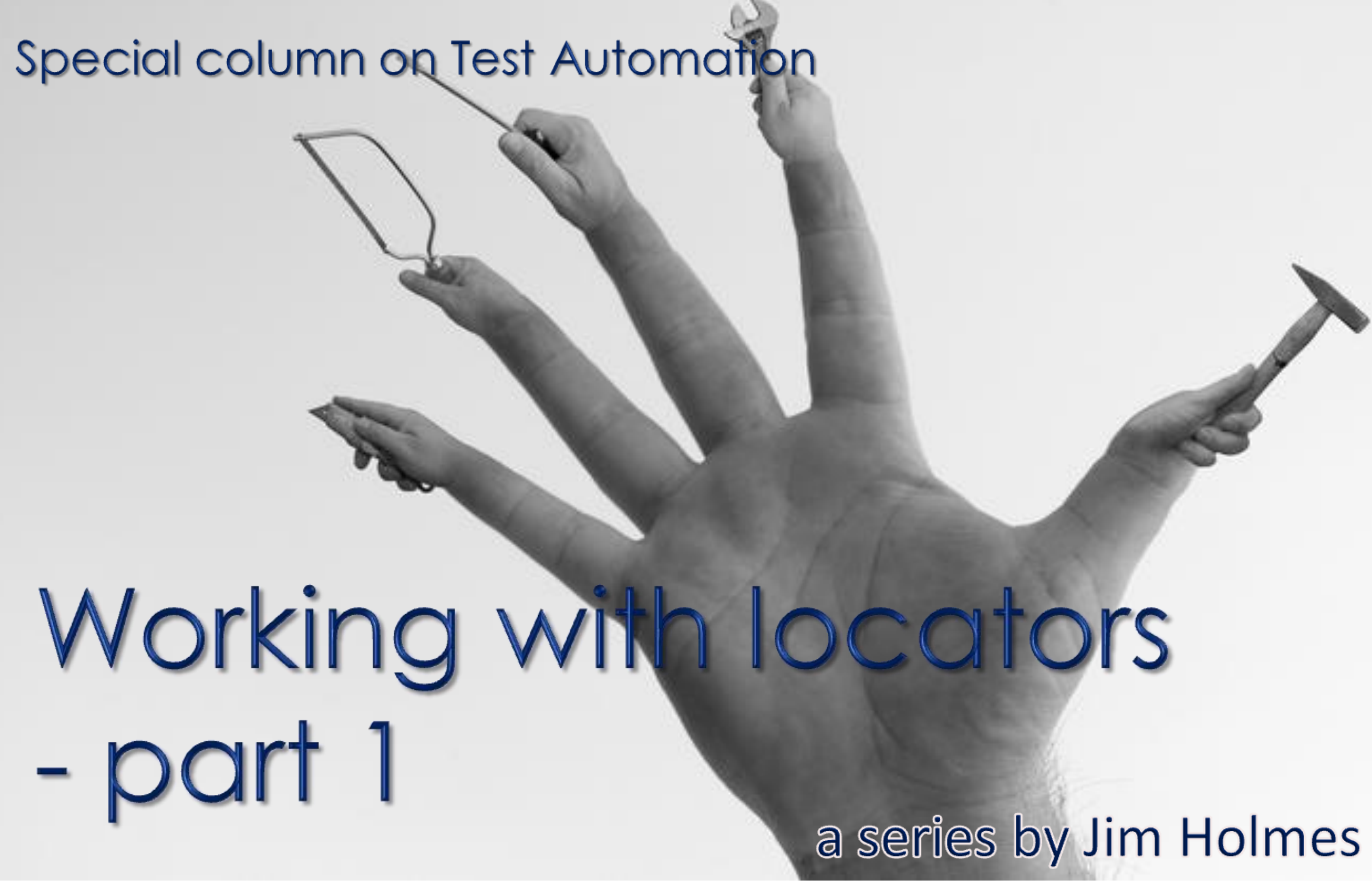
Please Note :

1. This is not a job portal.
2. Typical interview questions will not be answered.
3. Questions should be on Software Testing or related topics only.

Image courtesy : MMVI New Line Production

In the school of Testing
for your better learning & sharing experience

Special column on Test Automation



Working with locators - part 1

a series by Jim Holmes

Locators: Everything Hinges Upon Them

My previous two articles on user interface automation have all been philosophical, hand-wavy things. Now it's time to dive into some practical mechanics.

The most critical concept for UI automation is that of *element location*. It's called by several different names depending on what tool you're using or who you're talking to: find strategy, lookup, control identification, etc. The root concept is the same across all those vernaculars and the various tools. It's how that tool's scripts find the things on the page they need to interact with (or view, if you're working mobile or desktop automation).

Poor locators will drive your team crazy for a number of reasons:

- Tests fail when the target element moves on the UI
- Tests fail when the UI changes elsewhere (new elements, elements deleted, elements moved, etc.)
- Tests fail when data

This article will walk through some fundamental concepts of how locators work. I'll focus on web automation; however, the concepts are the same for other types of UI technologies.

One thing that's critical to learn about locators: the strategies you learn on one project will change on the next. Locators are impacted by a tremendous number of variables:

- Stakeholders. They're the ones defining how the system works.
- The technology stack you're working with. ASP.NET creates web pages differently than Ruby on Rails, for example. A number of different libraries and tools on top of those can change how the UI is rendered.
- Controls. Regardless of whether you're using commercial, open source, or in-house controls, you can bet they've their own approach to rendering.
- Designers. They do all this crazy layout and styling stuff that impacts how the DOM and page are structured.
- Developers. They've got to turn everything from the stakeholders, designers, and others into stuff that actually works. Their approach to building the pages has an extraordinary impact on how testable the UI is. They can be your worst nightmare. They can also be incredible allies who can give you lots of help in creating testable UIs. You choose how you want to relate to them…

Looking at the list above it's easy to see how locator strategies will vary when you change projects. The best thing you can do is learn solid generalized approaches for working with locators, then learn how to adapt them when you move to another project.

Storing and Defining Locators

How you store your locators in your test scripts is critical. You absolutely must use some form of centralized definition for your locators. Avoid, at all costs, duplication of locator definition. Duplicated locators means you'll have to edit ten, 50, or hundreds of files every time your UI changes. Note that's *when*, not *if* your UI changes… That's simply untenable over the long run. You'll end up spending more time fixing broken tests than adding value elsewhere, and it's guaranteed you'll miss updating elements. Guaranteed.

There are several approaches to locator storage: dictionaries, static classes, and the page object pattern are just a few.

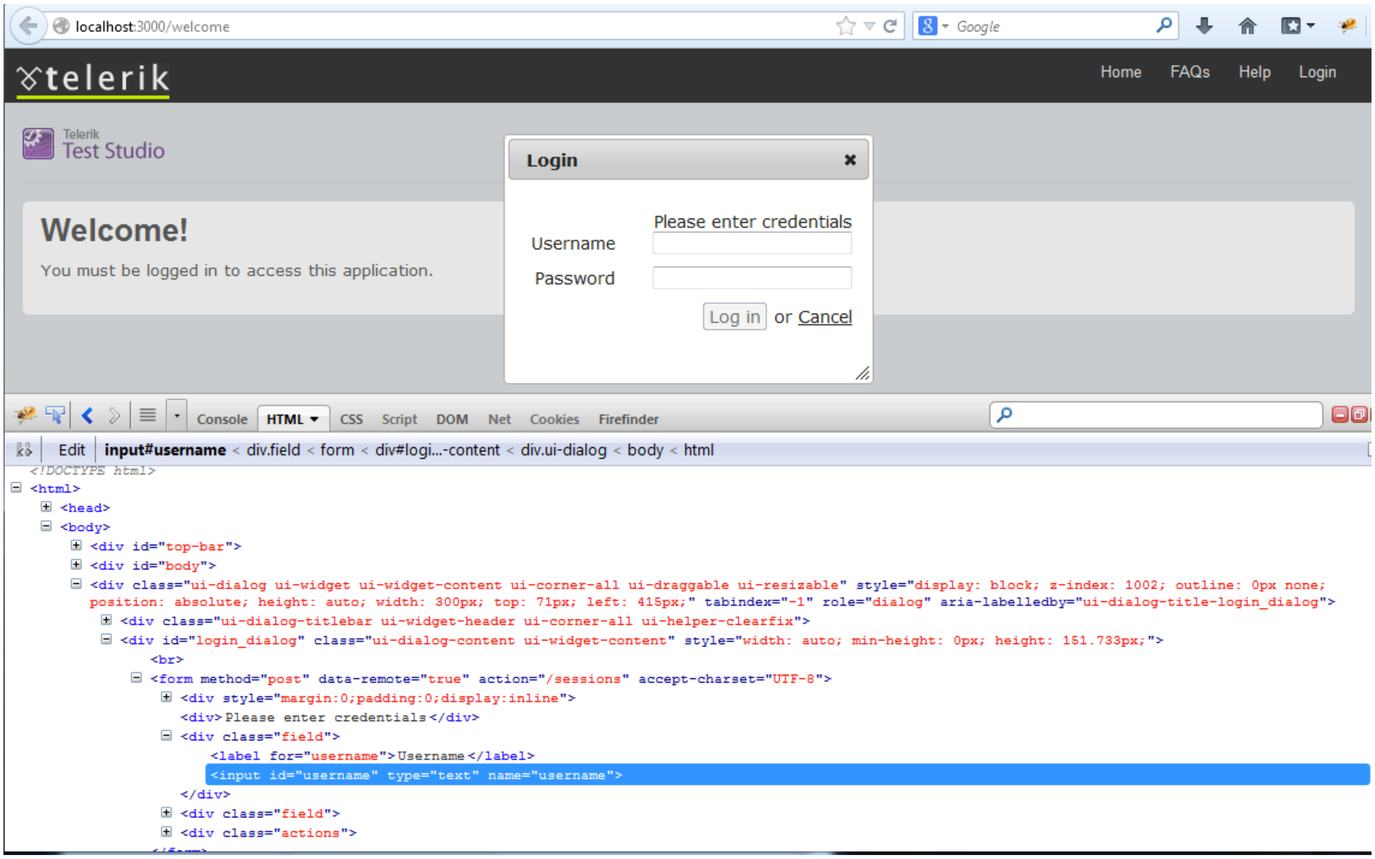
In this article I won't be using any of these. I'm purposely leaving the definitions inline for clarity's sake. Sometimes you do the wrong thing (or less-optimal-thing) when writing demo code, and this is one of those instances.

I'll be covering page objects in a later article, but for now please remember: Kids, don't do this in your production code!

How Locators Work

Locators give your test script a way to find objects of interest on the page or view. Locators rely on the structure of the page, properties of the target element, or sometimes both.

Let's have a look at the Document Object Model (DOM) of a simple web page. Let's imagine we're writing a logon script that will be used as a component for other tests. In this case we'll need to click the Login link, then fill in user name and password, and finally click the Login button.



Welcome!

You must be logged in to access this application.

Please enter credentials

```
<!DOCTYPE html>
<html>
  <head>
  <body>
    <div id="top-bar">
    <div id="body">
    <div class="ui-dialog ui-widget ui-widget-content ui-corner-all ui-draggable ui-resizable" style="display: block; z-index: 1002; outline: 0px none;
    position: absolute; height: auto; width: 300px; top: 71px; left: 415px;" tabindex="-1" role="dialog" aria-labelledby="ui-dialog-title-login_dialog">
      <div class="ui-dialog-titlebar ui-widget-header ui-corner-all ui-helper-clearfix">
      <div id="login_dialog" class="ui-dialog-content ui-widget-content" style="width: auto; min-height: 0px; height: 151.733px;">
        <br>
        <form method="post" data-remote="true" action="/sessions" accept-charset="UTF-8">
          <div style="margin:0;padding:0;display:inline">
          <div>Please enter credentials</div>
          <div class="field">
            <label for="username">Username</label>
            <input id="username" type="text" name="username">
          </div>
          <div class="field">
          <div class="actions">
        </form>
```

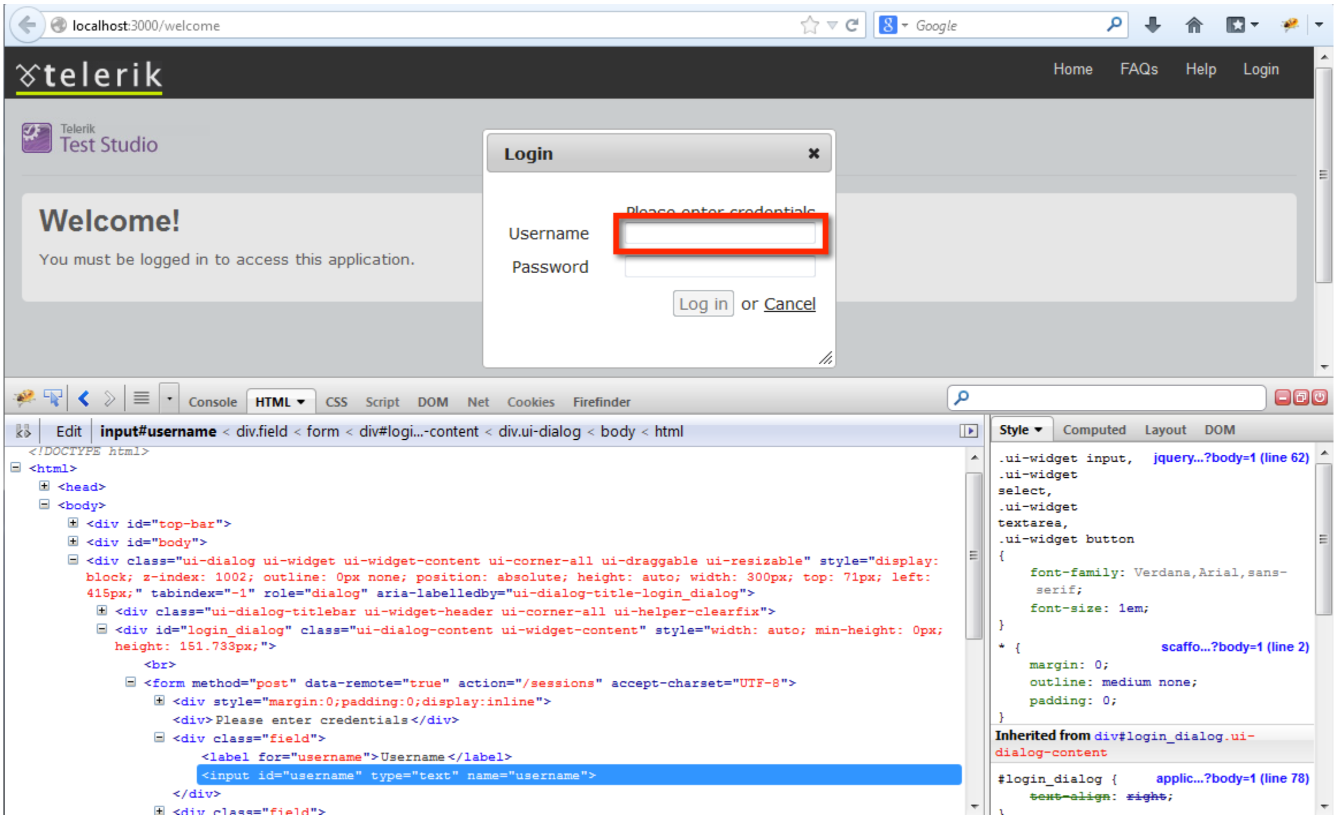
The image above shows part of the page's structure including the elements for the Username label and input fields. The rest of the article will focus on this example as we walk through different approaches for locating elements.

Unsurprisingly, locator mechanics differ between automation toolsets, and can also differ in how teams use those toolsets.

Types of Locators

Locators can be defined in several ways. You'll absolutely need to know several different approaches for your project—it's rare that every element in your page/view can be accessed with the same strategy.

We'll use the following screen as an example when walking through discussing the various locator types. We'll focus in on locating the input element for the Username field—the one that's highlighted on the page below and in the DOM displayed in the Firebug plug-in for Firefox.



You must be logged in to access this application.

```
<!DOCTYPE html>
<html>
  <head>
  <body>
    <div id="top-bar">
    <div id="body">
    <div class="ui-dialog ui-widget ui-widget-content ui-corner-all ui-draggable ui-resizable" style="display:
    block; z-index: 1002; outline: 0px none; position: absolute; height: auto; width: 300px; top: 71px; left:
    415px;" tabindex="-1" role="dialog" aria-labelledby="ui-dialog-title-login_dialog">
      <div class="ui-dialog-titlebar ui-widget-header ui-corner-all ui-helper-clearfix">
      <div id="login_dialog" class="ui-dialog-content ui-widget-content" style="width: auto; min-height: 0px;
      height: 151.733px;">
          <br>
        <form method="post" data-remote="true" action="/sessions" accept-charset="UTF-8">
          <div style="margin:0;padding:0;display:inline">
          <div>Please enter credentials</div>
          <div class="field">
            <label for="username">Username</label>
            <input id="username" type="text" name="username">
          </div>
          <div class="field">
```

```
.ui-widget input,     jquery...?body=1 (line 62)
.ui-widget
select,
.ui-widget
textarea,
.ui-widget button
{
    font-family: Verdana,Arial,sans-
     serif;
    font-size: 1em;
}
* {                    scaffo...?body=1 (line 2)
    margin: 0;
    outline: medium none;
    padding: 0;
}
Inherited from div#login_dialog.ui-
dialog-content
#login_dialog {      applic...?body=1 (line 78)
    ~~text-align~~: ~~right~~;
}
```

IDs

An *ID* is a value for an attribute in an HTML element. IDs aren't required to be present, but can be generated by the platform stack, the control, libraries, or manually added/altered by the developer. If the page is valid HTML then every ID value must be unique across the entire page. Please note I explicitly said *valid HTML*. Invalid HTML pages might have duplicated IDs.

Using IDs in WebDriver is extremely simple as demonstrated by the following snippet of C# code which uses the browser object to find elements by their ID value and perform various actions on them:

```
browser.FindElement(
    By.Id("login_link"))
    .Click();
browser.FindElement(
    By.Id("username"))
    .SendKeys("testuser");
browser.FindElement(
    By.Id("password"))
    .SendKeys("abc123");
browser.FindElement(
    By.Id("login_button"))
    .Click();
```

IDs as locators bring us a couple tremendous benefits. First, because each ID value must be unique, our script can find that element regardless of changes to the page. Our element can move around, other elements can be added or removed on the page, but our script will always find the element we're looking for.

There's also a critical issue to consider about IDs: *speed*. Every browser resolves IDs lightning fast. Microseconds may not seem like much individually, but it can add up to significant blocks of time when you're looking at a test suite with tens of thousands of tests and huge numbers of element resolutions.

With all these advantages, you'd think I'd say "It's a Best Practice to always use IDs!" You'd be wrong. First off, there's no such a thing as Best Practices (other than to use your brain) and secondly, there are times when using IDs will make your tests more brittle.

IDs Can Lead You Astray

Focus on always using IDs and you'll run into failing tests immediately where those IDs are generated dynamically, meaning they change based on environmental conditions. Several factors may cause IDs in your system to be dynamically generated:

- Your platform. ASP.NET Webforms, for example, create IDs for controls based on their hierarchy in the control tree. Change the tree and all controls get new IDs.
- Your controls. Many controls, regardless of who makes them, generate IDs automatically for data they present—like rows in a grid.
- Your UI developers. Developers make many choices in constructing the systems. Some of those may impact how IDs get generated.

IDs are generally your friend. Know when to use them, but also keep a weather eye for situations in your testing where they're not appropriate.

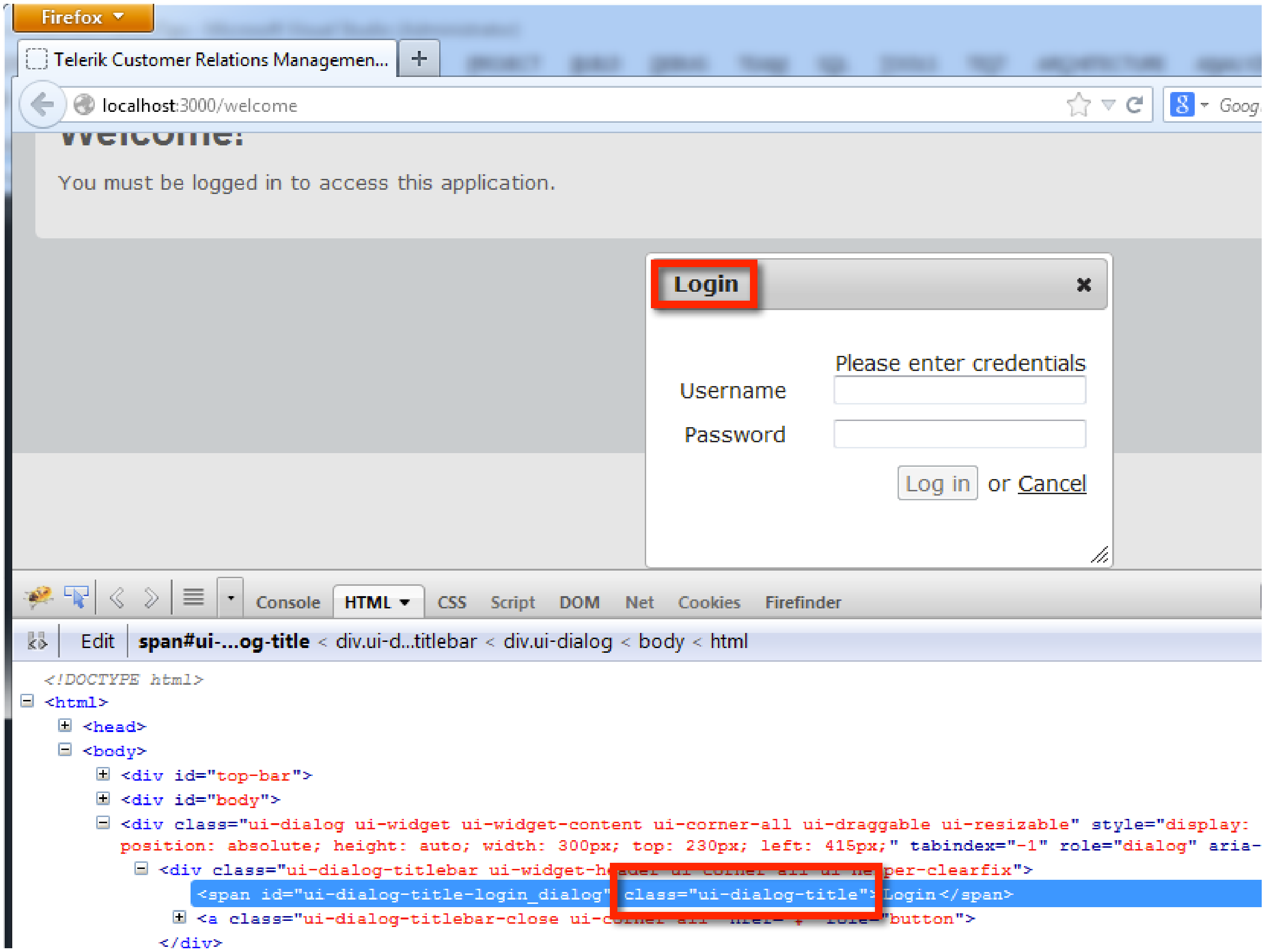
CSS

Cascading Style Sheet (CSS) *class* attributes can be used in a number of ways for element location.

First, you're able to directly use the class attribute itself as a locator. Secondly, you can use JQuery-style selectors to find your elements.

CSS class attribute values don't have to be unique. They rarely are since they're intended to provide reusability of style definitions. That said, sometimes they're unique enough [1]to be used for the particular test you're building.

The Input field we've been looking at above doesn't contain a *class* attribute, so we'll use the Login title of the modal that pops up for an example:


Firefox
Telerik Customer Relations Managemen...
localhost:3000/welcome
Welcome!
You must be logged in to access this application.
Login
Please enter credentials
Username
Password
Log in
or Cancel
Console
HTML
CSS
Script
DOM
Net
Cookies
Firefinder
Edit
span#ui-...og-title < div.ui-d...titlebar < div.ui-dialog < body < html

```
<!DOCTYPE html>
<html>
  <head>
  <body>
    <div id="top-bar">
    <div id="body">
    <div class="ui-dialog ui-widget ui-widget-content ui-corner-all ui-draggable ui-resizable" style="display:
    position: absolute; height: auto; width: 300px; top: 230px; left: 415px;" tabindex="-1" role="dialog" aria-
      <div class="ui-dialog-titlebar ui-widget-header ui-corner-all ui-helper-clearfix">
          <span id="ui-dialog-title-login_dialog" class="ui-dialog-title">Login</span>
        <a class="ui-dialog-titlebar-close ui-corner-all" href="#" role="button">
      </div>
```

CSS locators are nearly as fast as IDs when resolving in all browsers, so you don't have to worry about speed impacts when rolling along with them.

[1] This brings up the important concept: locators don't have to be perfect. They have to be good enough for the task at hand. Perfection is the enemy of getting stuff done!

CSS Selectors

CSS selectors enable you locate an element based on JQuery-like selector patterns. These selectors create a unique locator by piecing together values, element types, locations, and similar other items.

Below is a snippet showing the same C# code we used earlier for demonstrating IDs, but this time using CSS selectors:

```
browser.FindElement(
    By.CssSelector("a[id^='login']"))
    .Click();
browser.FindElement(
    By.CssSelector("label[for='username']+input"))
    .SendKeys("testuser");
browser.FindElement(
    By.CssSelector("label[for='password']+input"))
    .SendKeys("abc123");
browser.FindElement(
    By.CssSelector("input[value='Log in']"))
    .Click();
```

CSS selectors are fast to resolve, very flexible, and are fairly understandable to read. Unfortunately CSS selectors can only walk down the DOM; they can’t be used to move from one part of the DOM up and over to another area.

For that you’ll need our next locator type: XPath.

XPath

XPath may be one of the most abused, misunderstood, and outright hated technologies created in modern times. Perhaps ever. It contends with SharePoint and bad opera for the top place on my personal list of all things hated.

XPath is an extraordinarily powerful tool that enables you to inspect values of elements; navigate up, across, and down the DOM; and create extremely powerful locator strategies.

It can also be extremely brittle and high-maintenance if poorly used. Additionally, XPath is the slowest locator strategy for all browsers, especially Internet Explorer.

Overly brittle XPath locators are those which rely on starting too high in the DOM, or which rely on risky indexing. Returning to our username input field, a poorly formed XPath might look like this:

```
/html/body/div[3]/div[2]/form/div[2]/input
```

Two main problems impact this XPath: it starts from the document's root HTML element, and it uses fixed indexes to select which specific element from a group to use, e.g. div[3]/div[2]. This means nearly any change to the document's structure will break the locator and cause the test to fail.

Instead of that messy XPath, let's start from the input field we need, and look up and around the DOM for something unique that's close by. The text label with Username will serve well for us! From there we can move up one level, over to the sibling input label and back down. We can now use a much simpler, much more flexible XPath of:

```
//label[text()='Username']/../input
```

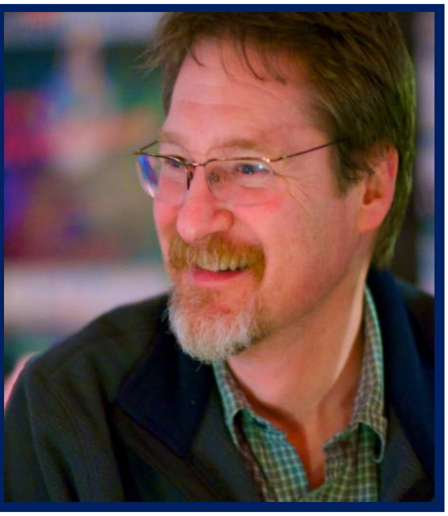
This locator also shows XPath's ability to walk up, over, and down the DOM to create complex locators.

XPath is a tool. Used with careful thought it's a great tool. Used with reckless abandon it's a recipe for lots of pain.

To be continued in next issue…

Jim Holmes is the Director of Engineering for Test Studio at Telerik. He has over 25 years in the IT field in positions including PC technician, WAN manager, customer relations manager, developer, and yes, tester. Jim has held jobs in the US Air Force, DOD sector, the software consulting domain, and commercial software product sectors. He's been a long-time advocate of test automation and has delivered software on a wide range of platforms.

He co-authored the book Windows Developer Power Tools and blogs frequently at http://FrazzledDad.com. Jim is also the President of the Board of Directors for the CodeMash conference held in the middle of winter at an indoor waterpark in Sandusky, Ohio.



Back To Index

Rants & Ramblings of a Mobile Tester

- by JeanAnn Harrison

Device Configuration Testing with focus on User Experience

In the Mobile Software Apps world, immediate satisfaction is vital for any company developing mobile apps. What testers and developers can forget is testing for user experience and instead put too much focus on functions like buttons working as expected, or links working. But are mobile testers considering how the user actually uses the mobile app based on what device they are using that app?

Recently I attended STPCon in Phoenix, AZ and was fortunate enough to meet Carlo Cadet from Perfecto Mobile, Inc. His session title: "How to Build an Enterprise Grade Mobile Testing Strategy" which is quite a mouthful. Carlo talked about how companies need to come up with a mobile strategy which makes sense for the company and company culture. One quote really stuck with me, "what people do on a phone is research while those who have a tablet also, use it to make purchases online." So from a User Experience point of view, with a short timeline on the mobile testing project, what kinds of tests would you put focus on? Would you apply the same test cases for both the tablet version and the mobile phone version? If so, I would love to hear about your reasoning. I will share some test case ideas in a bit but want to share some definitions and perspective which is important for any mobile tester to consider.

Have you thought about doing any Device Configuration Testing? Putting some extra focus on User Experience testing (keep in mind this is not Usability testing) considering the device type, and thinking only one set of test cases to one mobile device or one type of mobile device is nowhere near the level of acceptable test coverage if you want to maintain or increase your user/customer base. So you're asking yourself "how does she KNOW this" and I can tell you… I've done the testing. I've learned through mistakes and changed my approach.

Let's start our discussion, though, with defining "User Experience" as well as "Device Configuration Testing". Dr Philip Lew, CEO of XBoSoftInc regularly speaks about mobile user experience, defining it as a more broad term than "usability" which is a more detailed point of view of the test type. I think of User Experience as an umbrella, if you will, over terms like: useful, findable, accessible, desirable, valuable, credible and usable. If a mobile application crashes but recovers in a fast, non-intrusive way for the user, then we're talking desirable. I would apply performance tests on this situation which is very different than being able to find the correct icon to function as expected which is more of a usability concept.

User Experience for Mobile is the overall experience of any experience a user feels while using any mobile device including phones, tablets, and proprietary devices. In some instances, you could say laptops & notebooks are mobile devices too. Have you thought about testing your laptop the same as your mobile phone, meaning, does your laptop perform in the same way as your mobile phone when using a specific application?

Before jumping in the pool of 'let's test", we need to really think about planning out which types of tests apply to which device. A great way to learn more about Device Configuration Testing is to "test it out" with your own devices. For example, have a desktop version of a popular software application where you know has a mobile version you can download to your other devices. If we take Twitter, don't we want to think about how we ourselves use Twitter first? Are you like me and use Twitter on your phone, tablet AND the laptop? I do this for convenience but I also do this so I can SEE for myself what differences exist for one application being used on all three different devices. Very interesting results. What is even more of an eye opener are the differences in usage from comparing an iPhone and an Android tablet and/or Windows (for the desktop). Unlike testing desktop applications, you're primarily testing for one operating system, the mobile operating systems all must be included in standard testing. This is only level of complexity to your mobile testing project. After all, the iPad version of Twitter versus the iPhone version versus the Android tablet version all have different displays including text size, fonts, colors, brightness to name a few factors.

Do mobile testers take the tie to compare these factors on different device configurations? Is it important for your company's reputation to do so? Maybe testers being asked to participate in the mobile testing projects do not realize these tests are important because they do not realize there is a direct impact on the bottom line, short and long term. Does management allow time within the mobile project to allow for such testing? If not, testers need to educate stakeholders by showing how User Experience tests are vital to the company's viability.

Let's think about strategy then, to become more efficient in our mobile testing projects, delivery of mobile software applications where companies can have higher assurance of User Experience test coverage. Here are some questions to ask:

1. What kinds of customers will be using your application? I'm talking general demographics here, age, gender, location throughout the world,

2. Will you have a free version to download and a cost based application with more features?

3. What kind of features do you expect to have a greater impact on usage?

4. What kinds of features are your customers willing to pay a fee to have on their mobile device?

5. What is the likely place or set up the customer will be engaging with the application? For example, with the customer be at home, office, walking/driving etc.

6. Are your customers' technically savvy? Does your mobile application require any special technical knowledge to get the best user experience?

7. Which device is your customer base primarily using as their device of choice in using your application?

8. Would the customer likely switch from device type to device type depending on convenience or perhaps a specific purpose as Carlo stated?

9. Will customers be relying on the mobile website version of the product or will they be using the hybrid application on the phone? Which device would they likely use for each situation?

Knowing as much as you can about how your customers use or will likely use the application on a particular device and how often will give you a solid start to developing your device configuration testing strategy. Of course this might mean the mobile test team actively working with the marketing and sales teams to learn more about the customer base. A great way for the company to learn about their customer base is to occasionally put out survey blog posts.

Let's revisit Carlo's comment about how a phone is primarily being used for researching while a tablet is likely to be used more for purchasing online surfing online for products, and consider what kinds of tests apply for our application under test. If we take Amazonas our application under test, would we be concerned about creating tests where we would spend a lot of time testing the back and forward buttons? Doesn't it depend on what version of the application they are using? Meaning are they using the mobile website version within a browser window or are they using the hybrid application they downloaded onto their phone?

What other tests can we consider to test the User Experience based on what device is being used for testing? Let's go back to that User Experience list of test types and define a bit more detail for each test type.

1) Useful – Is it useful to have the same size font displayed for both the tablet and the phone? Or would you want to test to see how useful the chosen font size works for the tablet and how the font size displays on the phone? Let's also consider various phone sizes and corresponding displays. Bottom line: is the display useful on the phone and the tablet and any other device where the application might be used?

2) Accessible - what features are accessible on each page of the application under test? Is there a sequence which might make the User Experience more successful based on the type of device being used? Mobile phones come in different sizes as we have small one hand operational type phones as well as those large phones and not quite tablets, which require 2 hands to use properly.

3) Findable – On my desktop version of the Search function is easy to find for the Facebook web application. But if I look at my mobile phone hybrid version, I don't see it on the default home page. Yet the tablet version does display the search function on the mobile hybrid application. Is this an important function

to my user/customer base? Maybe but as a company trying to make sure the user experience is a comfortable one, and perhaps the tablet version is more likely to upgrade to a paid version of the application, test cases should probably have a stronger focus than on the phone version.

4) Valuable – Is it valuable for the customer to have one functionality over another and then based on a certain device type? Going back to my FaceBook example, is it more valuable to have the search function removed from the home page on a phone display but not on the tablet? What does valuable mean to the customer and to the application itself? What about saving on resources? Would the customer find more value in NOT displaying the search feature on the home page so resources could be saved and used in another way?

User Experience testing based on the device configuration is a testing strategy which is often missed. Yet, with the user experience being vital to a company's ability to survive in the mobile app world, how can this type of testing not considered vital to every mobile project? I often use this phrase: "Plan Your Tests" and do not forget, when the project's testing timeline is short, put focus on User Experience Testing. Remember how you use your configuration types, switching from one device to another. What are your own personal expectations in using your devices and apps? Make sure your customers have an immediately fantastic experience using your mobile application no matter the device in which they use that application.

To be continued in next issue…

Jean Ann has been in the Software Testing and Quality Assurance field for over 14 years including 6 years working within a Regulatory Environment and 7 years performing mobile software testing. Her niche is system integration testing with focus multi-tiered system environments involving client/server, web application, and standalone software applications. Mobile software testing includes mobile native apps, mobile hybrid apps, mobile web applications and mobile websites.

Jean Ann is a consistent speaker at many software testing conferences, a Weekend Testing Americas facilitator as well as making guest appearances. She is always looking to gain inspiration from fellow testers throughout the software testing community and continues to combine her practical experiences with interacting on software quality and testing forums, attending training classes and remaining active on social media sites.

Back To Index

Introducing Change in your Testing Department

a series by Bernice Ruhland

Introducing Session-Based Testing To Your Team

Part 1 provided basic definitions of session-based testing and other terminology that will be used in this series. This article is focused on introducing session-based testing in your testing group after attending training.

My Background

To provide some background, I learned about session-based testing (SBT) during the early 2000's. Last year I attended two remote Rapid Testing Intensive (RTI) seminars. It was wonderful to better understand an approach I studied and have it brought to life by those who developed it. Over the years, I have attended many workshops and training programs. RTI is unique because it is not the typical cookie cutter approach to training. Instead James Bach uses a blended learning model providing hands-on assignments that applies to a tester's work life. Prior to attending the seminar I used SBT in a limited fashion. After the seminar I had more ideas on how to use it as a testing approach on a regular basis. You may wonder why I attended a second seminar - it was to continue practicing. James and other testing experts provide feedback on select assignments, which leads to a great learning experience.

Using Sessions Right after Training

After attending training, I created a survey session to test an enhancement being added to our product. A charter was not created for the intake and setup sessions; however we went through the process. Before creating the survey charter, we understood the goal and deliverables. Any provided documentation was reviewed. From there we made sure we had the correct test data and test users defined. The survey session provided an opportunity for the tester to learn more about the enhancement. The debriefing session provided valuable feedback to the testing team to understand what was available for testing, how it was the same or different from similar functionality, and to determine next steps. The positive feedback using a survey session was valuable in gaining interest from other testers.

I would recommend to start using sessions as soon as you can. Similar to the above example you can select any testing problem and it does not need to be a large project. Sometimes when introducing change it is good to start with something small. The team learns on a smaller scale and may be able to use different types of charters in a shorter time period. Sometimes introducing a new approach for testing a complex project may receive resistance from the team. In this situation, consider if SBT could be used for a portion of the project. For example, a survey charter could be used to find initial problems and help the team learn. Basically, you know your team and how easily they adapt to change. If you have early adopters or champions in your team, try to work with them first so they can be a change agent influencing others to use sessions.

Debriefing Meetings

A debriefing session is a great way for a two-way conversation on what was learned during testing and to determine next steps. This may be a new approach for many people especially since the debriefing should be a short meeting. Consider rotating the tester and reviewer role allowing everyone to understand the responsibilities of both roles and the meeting's value. I believe experiencing both roles provide hands-on knowledge to prepare for the meeting. For example, if I experience how a reviewer reacts to information can help me better prepare when I am presenting testing results.

Sometimes we are asked why we need a debriefing session because it is not the best use of time. An argument we often hear is that it would be better to stay focus on getting through more charters. The debriefing session is an opportunity to have a short conversation to discuss what was witnessed in testing to better understand the potential problems and challenges we are encountering. The information is fresh and can uncover new testing ideas. Plus I would rather address problems and challenges now instead of later when there is a potential of running out of testing time. Debriefing sessions with new testers can be an opportunity to provide feedback on their testing approaches and general testing guidance. A debriefing session typically should not be skipped. However, with more experienced testers it might be appropriate to have them complete a few charters before meeting.

You do not need to use the same approach with each tester; however, do not completely eliminate them either. Find the right balance and do not be afraid to make changes based upon what is best for your team.

Using Sessions to Train

Another option is to use sessions to train on existing functionality. For the more mature parts of the product it can be difficult for testers to know where to start. Working through different session types provide a focused yet exploratory approach to learning. For example during the Setup session there might be specific test data to load, test users to create with different permission levels. Understanding the setup is important but it is better to have hands-on training of loading the test data and creating the test users. The setup charter can provide time for that learning activity. When it is finished, have the tester complete a survey session and progress to other charters as appropriate. Hold a debriefing meeting after each session encouraging the tester to ask questions on expected behavior and review potential bugs discovered through the charter. These meetings can help the trainer determine the next charter to progress the learning experience. My experience with using this approach is successful whether it is training a newly hired tester or cross-training testers on other areas of the product. It is helpful to have a rough idea on how you are going to approach breaking out the charters. While at the same time staying flexible to make changes based upon the learning needs and progress of the tester. Typically I write one charter at a time and might sketch out ideas in a mindmap. Another option is to create a Product Coverage Outline to help both the tester and trainer ensure proper training coverage.

Bernice Niel Ruhland is a Director of Quality Management Programs for a privately-owned software development company. She provides strategic oversight and leadership of a Software Testing Department using context-driven and agile approaches and techniques. She participates in company-wide quality initiatives and programs.

To complete her Masters in Strategic Leadership, she conducted a research project on career development and onboarding strategies. The opinions of this article are her own and not reflective of the company she is employed.

You can connect with Bernice at:

LinkedIn http://www.linkedin.com/in/bernicenielruhland

Twitter: bruhland2000 or

Her blog: TheTestersEdge.com

Back To Index

Happiness is....

Taking a break and reading about **testing** !!!

Call for Articles !

Have you got something to say?

yes, we are listening you...!!!

"Tea-time with Testers" firmly believes that one of the best ways to improve upon software testing is to listen to the lessons learned by others and their experiences too.

So, if you have an interesting story that you'd like to share with the world, contact us at teatimewithtesters@gmail.com.

Submit your articles, stories, thoughts around software testing.

now its your chance to be heard...!

Click HERE to read our Article Submission FAQs !

sciencephotogallery

Joy of Testing - Being Mindful & Mindless

Activities are important, but outcomes matter. Intellect plays a vital role in ensuring that the activities we do are indeed effective and efficient to ensure good outcomes.

Outcomes are great, but will they deliver the promise to the customer? It is about keen observations and fine adjustments that ensure outcomes deliver the value. It is about being in the 'flow'.

Let us analyze and discuss. It is the wonderful time of the year to reminisce, reflect and look forward to a joyful year. To a year filled with 'joy of testing' and delivering value to customers and the community.

The act of testing consists of a variety of activities that we perform. Some activities require intelligent thinking whilst some are repetitive. Intellect i.e. intelligent thinking play a vital role in being effective. A logical scientific approach to understanding, hypothesizing what could go wrong, performing activities to prove/disprove and continuous observation to constantly adjust is what makes testing challenging. And this is the basis of Hypothesis Based Testing (HBT). Being logical/rational seems interesting, but is this good enough? Is it fun? Does this not make it cold and impersonal? We may perform activities that that are indeed effective, but will they deliver value to end users? Should we not be emotional and keep the end users in mind to deliver value? And more importantly enjoy testing?

In additional to the intelligent activities, testing also consists of repetitive ones, that are perceived boring. And as intelligent testers, what do we do? We create a process to make the repetitive activities more efficient. In effect, be mindless. And slowly monotony sets in and we get bored. And we hate the structure/rigor imposed by the process. We perceive it to be curtailing creativity. Does being mindless make you dumb and not creative? Let us look at this differently.

Mindless requires you to become empty, so that you can be in the present, the state of mindfulness. Being in state of flow. Sheer joy. I discovered this in endurance cycling. Endurance cycling is about doing real long distances - a few hundred(s) of kilometers at one go. Like any endurance activity, this is also not about sheer physical strength, but largely about mental strength. When you ride for hours together non-stop constantly looking forward to finishing the distance (goal), it becomes monotonous, boring and saps your mental energy and exhausts you physically. And takes the fun out of cycling. What I discovered is to look just in front of the wheel, not the mile markers, and just focus on the pedal rotations. To just focus on now. In the process, I emptied my mind and became mindful, only aware of now, not reflecting on the past and blissfully ignorant of the future. Being in the present. Time stopped. I was in the 'flow'. The state of being aware of only now, and this was bliss. Sheer joy. And this is when you realize your peak potential. And you are so tuned, so observant and react to the environment in an extremely agile manner and seemingly without effort. And this is so beautiful.

So when you build a process to make the repetitive things easier, or to ensure that we have clarity of what to do when, don't dismiss this as being non-creative and boring, think of this as making way for you to be uncluttered so that you can be mindful and enjoy testing, focus on end user outcomes and therefore deliver value.

Intelligent thinking is being rational, of being analytical, to process observations logically to ensure effective outcomes. Being mindful allows one to be in the present and enjoy the moment, to be very observant/ sensitive and process these observations logically, course correcting constantly and delivering value. So just be in the present and enjoy every moment.

Joy of testing - Being mindful and mindless. A feeling that is blissful. May you experience this more often in the New Year.

To the wonderful folks of this community, wish you a wonderful 2014. To a year of joy, fun and value.

T Ashok is the Founder & CEO of STAG Software Private Limited.

Passionate about excellence, his mission is to invent technologies to deliver “clean software”.

He can be reached at **ash@stagsoftware.com**

Back To Index

testing intelligence

- its all about becoming an intelligent tester

an exclusive series by **Joel Montvelisky**

Whoever you are, whatever you do, YOU ARE FIRED!!!

This is the best advice I got from a co-worker some years ago.

Let me explain…

Why New Year’s Resolution Lists don’t work

OK, so we are on the last day of the year once again.

It’s the time when we look back at what we did and review if we are happy with what we’ve achieved. Some of us will go even further and make a New

Year's resolution list with all the things we want to change and achieve during this year that's about to start, but…

When we get back to work on Jan 2nd and we want to start with our "new resolutions" we understand that we still need to continue handling the same projects we started a couple of weeks ago with the same people and the same attitudes we dealt with up to now.

Then someone calls us and complains about a decision you made two months ago.

And so, before you know it, you are back doing the same things in the same way and with the same attitude you did before.

And so your New Year's resolution list becomes a nice decoration to post on the wall of your office or cubicle.

My conclusion: New Year's resolution lists are not that productive to begin with.

Start Fresh – Quit your job NOW!

There is not much we can do about continuing handling the same projects and working with the same people we did up to now. Nor is there a way to stop being responsible for the stuff you did (good or bad) some months ago. This is just the way it is.

But imagine, just for a minute, what would happen if today was your first day on the job.

Remember that feeling of a *fresh start* and the excitement you had during your first days on the job? How good would it be to be able to get that feeling again today!

Then again, why not? Make a decision that *today*, with the new year, you will virtually quit your job and start fresh once again tomorrow!!

Communicate this to your team and to your co-workers – who cares if some of them will look at you as if you drank too much last night! Starting tomorrow all quarrels are behind us, our past wrong decisions are forgotten, and all of us get a chance to start new. We all get a fresh start.

Make new objectives, go ahead and ask your team to do the same.

The important thing is to approach your work as if this is a fresh start and whatever you did in the past has no effect in your future actions or decisions.

It is all in your mind and your attitude

Choosing to virtually quit your job and start today fresh is up to you.

The guy who told me about this approach does this once a quarter, and even though at the beginning it used to look awkward, with time I understood what he did and even adopted this approach myself (although I only do it once a year and not once every three months).

Go ahead and give this a try! It will be even more interesting if you try it for a couple of weeks and then come back to share your experience.

What do you have to lose? 😉

Joel Montvelisky is a tester and test manager with over 14 years of experience in the field.

He's worked in companies ranging from small Internet Start-Ups and all the way to large multinational corporations, including Mercury Interactive (currently HP Software) where he managed the QA for TestDirector/Quality Center, QTP, WinRunner, and additional products in the Testing Area.

Today Joel is the Solution and Methodology Architect at PractiTest, a new Lightweight Enterprise Test Management Platform.

He also imparts short training and consulting sessions, and is one of the chief editors of ThinkTesting - a Hebrew Testing Magazine.

Joel publishes a blog under - http://qablog.practitest.com and regularly tweets as joelmonte

Back To Index

by

Quality is delighting customers
QUALITY
TESTING

Testing PUZZLES by Sebi

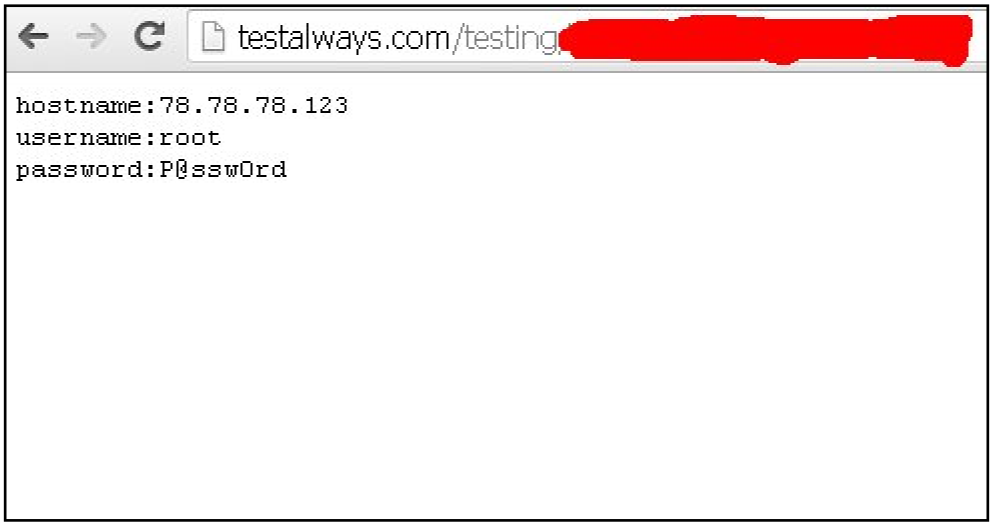
Claim your Smart Tester of The Month Award. Send us your answer for Puzzle b4 25th Feb 2014 & grab your Title.

Send -> editor@teatimewithtesters.com with Subject: Testing Puzzle

Puzzle

Find the entire URL that starts with 'http://testalways.com/testing/' where the configuration file in the attached picture is located.

Please test for this puzzle only inside http://testalways.com/testing/ and not other folders.



```
hostname:78.78.78.123
username:root
password:P@ssw0rd
```

Back To Index

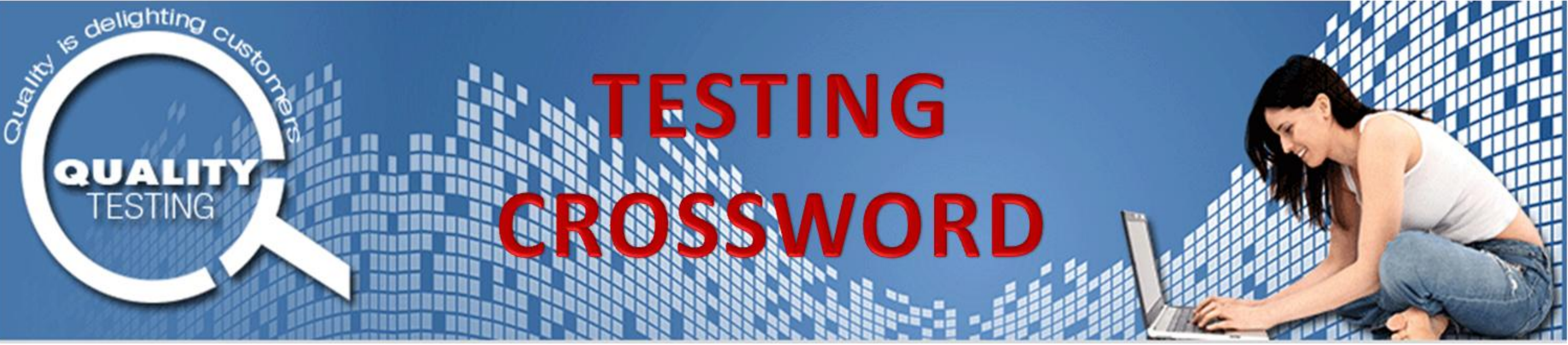
Biography

Blindu Eusebiu (a.k.a. Sebi) is a tester for more than 5 years.

He considers himself a context-driven follower and he is a fan of exploratory testing.

He tweets as @testalways.

You can find some interactive testing puzzles on his website www.testalways.com



TESTING CROSSWORD

1		2		3			4
	■		■		■	■	
	■	5		6			
7			■		■	■	
	■	8			9	10	
	■		■		■		
11							■

Horizontal:

1. Most wanted open source tool for web applications (8)

5. It is a testing conducted within the organization (5)

7. Continuously raising an input signal until the system breaks down, in short form (2)

8. Testing of individual software components (4)

9. It is collection of all the hardware, software, tools, test data, documents, Test scripts and other support elements required for conducting test, in short form (2)

11. Testing software through executing it (7)

Vertical:

1. Head of Marketing of TestPlant Company (7)

2. It is an open source and free load testing tool (6)

3. A computer system to analyze, understand and generate natural human languages. It is known as ______________________, in short form (3)

4. It is a software testing technique in which the testing is performed on the system under test randomly. It is called ___________________ testing (6)

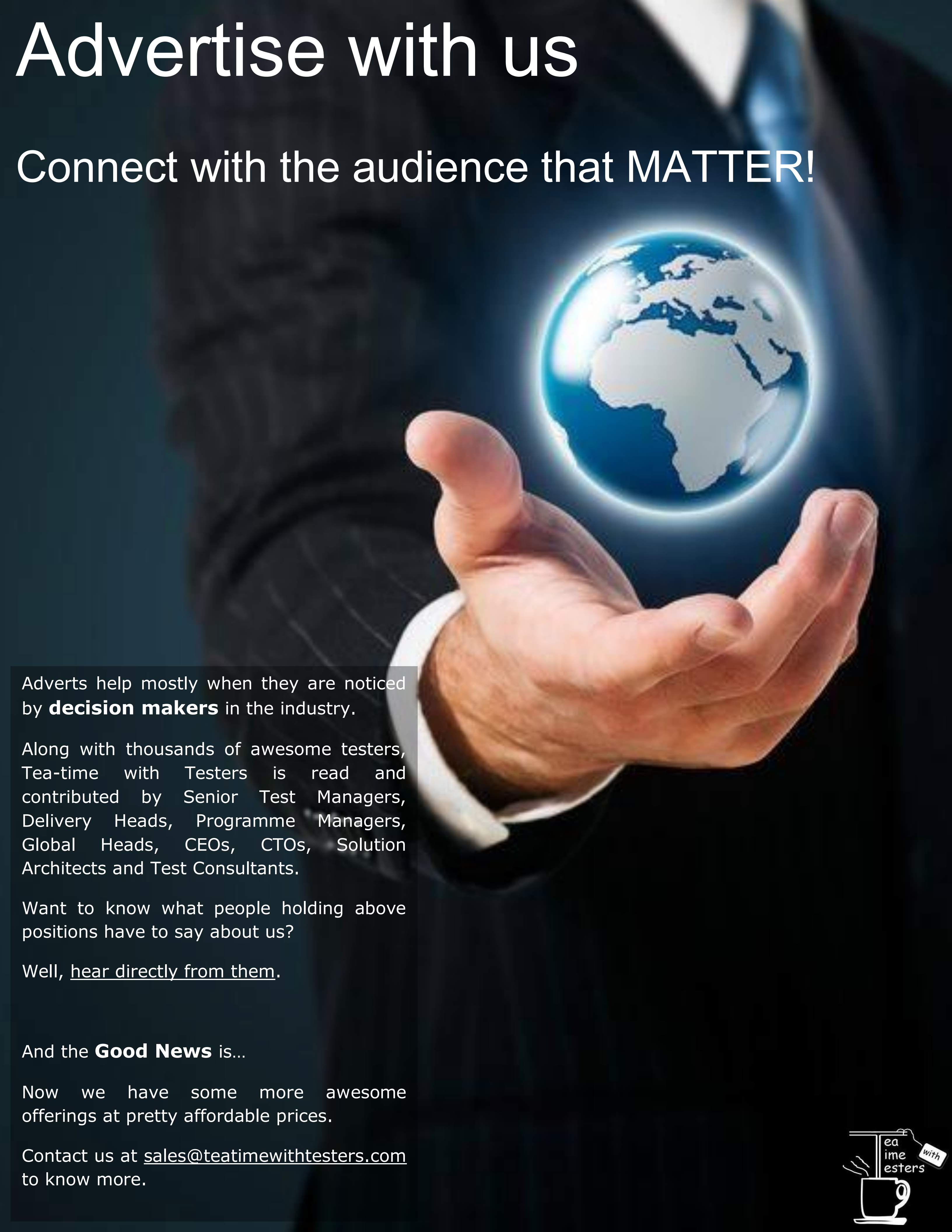
6. It is a way of approaching a test design process by having two people test the same thing at the same time and place, continuous exchanging ideas. It is known as ______ testing (4)

10. A test suite that exercises the full functionality of a product but does not test features in detail, in short form (2)

Click here for answers to last month's crossword

Advertise with us

Connect with the audience that MATTER!



Adverts help mostly when they are noticed by **decision makers** in the industry.

Along with thousands of awesome testers, Tea-time with Testers is read and contributed by Senior Test Managers, Delivery Heads, Programme Managers, Global Heads, CEOs, CTOs, Solution Architects and Test Consultants.

Want to know what people holding above positions have to say about us?

Well, hear directly from them.

And the **Good News** is…

Now we have some more awesome offerings at pretty affordable prices.

Contact us at sales@teatimewithtesters.com to know more.

Every Tester
who reads Tea-time with Testers,
Recommends it to friends and colleagues .
What About You ?

Image : vernhart

in nexxt issue

articles by -

Jerry Weinberg

T Ashok

Jim Holmes

Sanjay Gupta

JeanAnn Harrison

Joel Montvelisky

our family

Founder & Editor:

Lalitkumar Bhamare (Pune, India)

Pratikkumar Patel (Mumbai, India)

Lalitkumar

Pratikkumar

Contribution and Guidance:

Jerry Weinberg (U.S.A.)

T Ashok (India)

Joel Montvelisky (Israel)

Jerry

T Ashok

Joel

Editorial | Magazine Design | Logo Design | Web Design:

Lalitkumar Bhamare

Core Team:

Dr.Meeta Prakash (Bangalore, India)

Dr. Meeta Prakash

Testing Puzzle & Online Collaboration:

Eusebiu Blindu (Brno , Czech Republic)

Shweta Daiv (Pune, India)

Eusebiu

Shweta

Tech -Team:

Chris Philip (Mumbai, India)

Romil Gupta (Pune, India)

Kiran kumar (Mumbai, India)

Kiran Kumar

Chris

Romil

|| Karmanye vadhikaraste ma phaleshu kadachna |
Karmaphalehtur bhurma te sangostvakarmani ||

To get **FREE** copy ,

Subscribe to our group at

Google

Join our community on

facebook

Follow us on

www.teatimewithtesters.com

Give Feedback