2011

What If…

50+ tips to win testing contests

Think about it.

Ajay Balamurugadas

http://EnjoyTesting.blogspot.com

11/20/2011

What if…

Hello Reader,

Special thanks to you for downloading this book. This book is a small collection of tips based on my experiences in testing contests. I participated in a lot of contests at Test Republic, Software Testing Club, 99tests.com and contests on other testers' blogs or websites. After winning some of them, I realized that it can be easy to win a testing contest. All you need is some dedicated time, the right skills and practice.

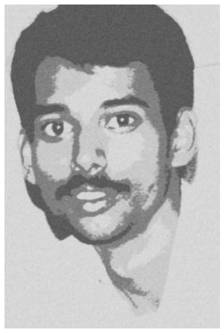
Weekend Testing (www.weekendtesting.com) and interactions with many other testers has helped me learn a lot in this industry. I have learnt a variety of test ideas by following the discussions on Twitter, reading testers' blogs and their challenges. Based on these experiences, I wrote this book. Participating in testing contests helped me improve my testing skills at work. The time pressure and competitiveness of testing contests has taught me many valuable lessons which can be incorporated in your day to day work.

What is your experience of participating in testing contests? After my first ebook - "What If…" which can be bought from http://bit.ly/mPjS3r, I hope that you enjoy this book too. Hope you enjoy the snippets while I wait for your comments…

Please feel free to contact me via email, Skype, Twitter or GTalk.

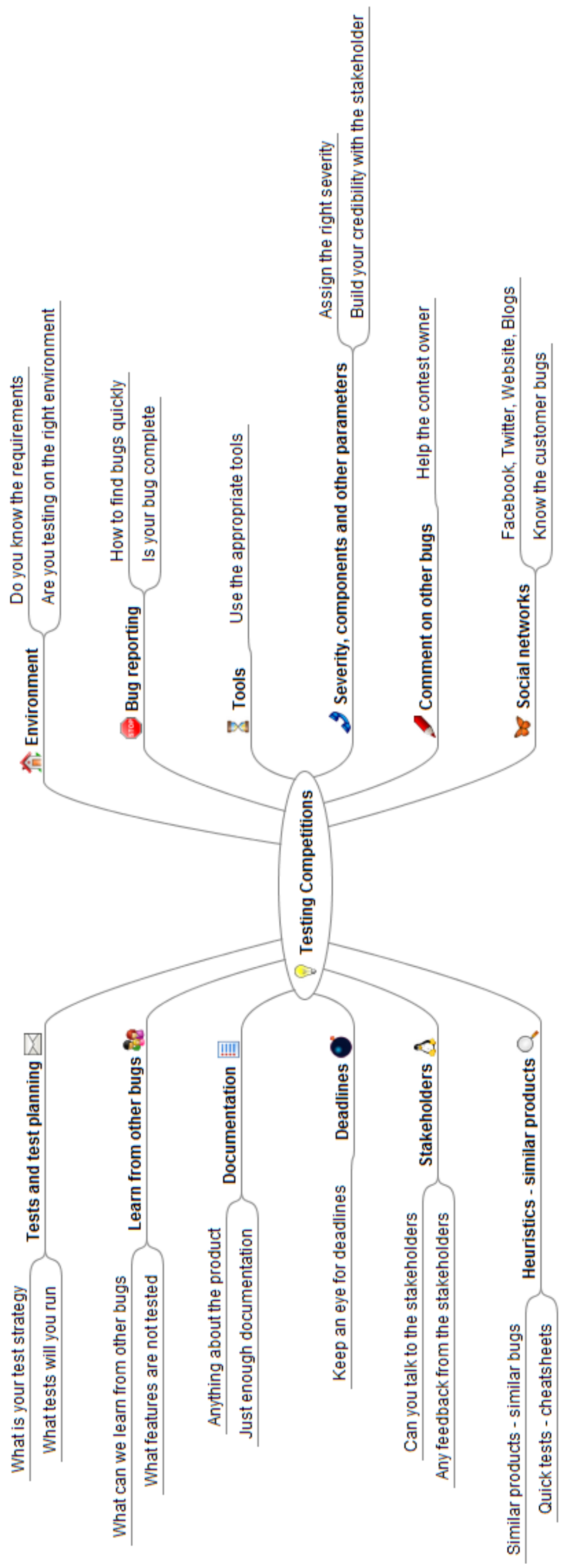
Email: ajay184f@gmail.com Blog: www.EnjoyTesting.blogspot.com

Skype/Twitter/GTalk: **ajay184f**



Contents

Testing Competitions Mindmap


Testing Competitions
Environment
Do you know the requirements
Are you testing on the right environment
Bug reporting
How to find bugs quickly
Is your bug complete
Tools
Use the appropriate tools
Severity, components and other parameters
Assign the right severity
Build your credibility with the stakeholder
Comment on other bugs
Help the contest owner
Social networks
Facebook, Twitter, Website, Blogs
Know the customer bugs
Tests and test planning
What is your test strategy
What tests will you run
Learn from other bugs
What can we learn from other bugs
What features are not tested
Documentation
Anything about the product
Just enough documentation
Deadlines
Keep an eye for deadlines
Stakeholders
Can you talk to the stakeholders
Any feedback from the stakeholders
Heuristics - similar products
Similar products - similar bugs
Quick tests - cheatsheets

Test Environment

One of the most important points to consider during testing is the **Test Environment**. It plays an even bigger role during testing competitions. Due to the time pressure in testing competitions, it is very easy and natural to forget about the test environment. As a tester, testing on the right test environment will ensure that your bugs are not marked invalid even before the validation starts.

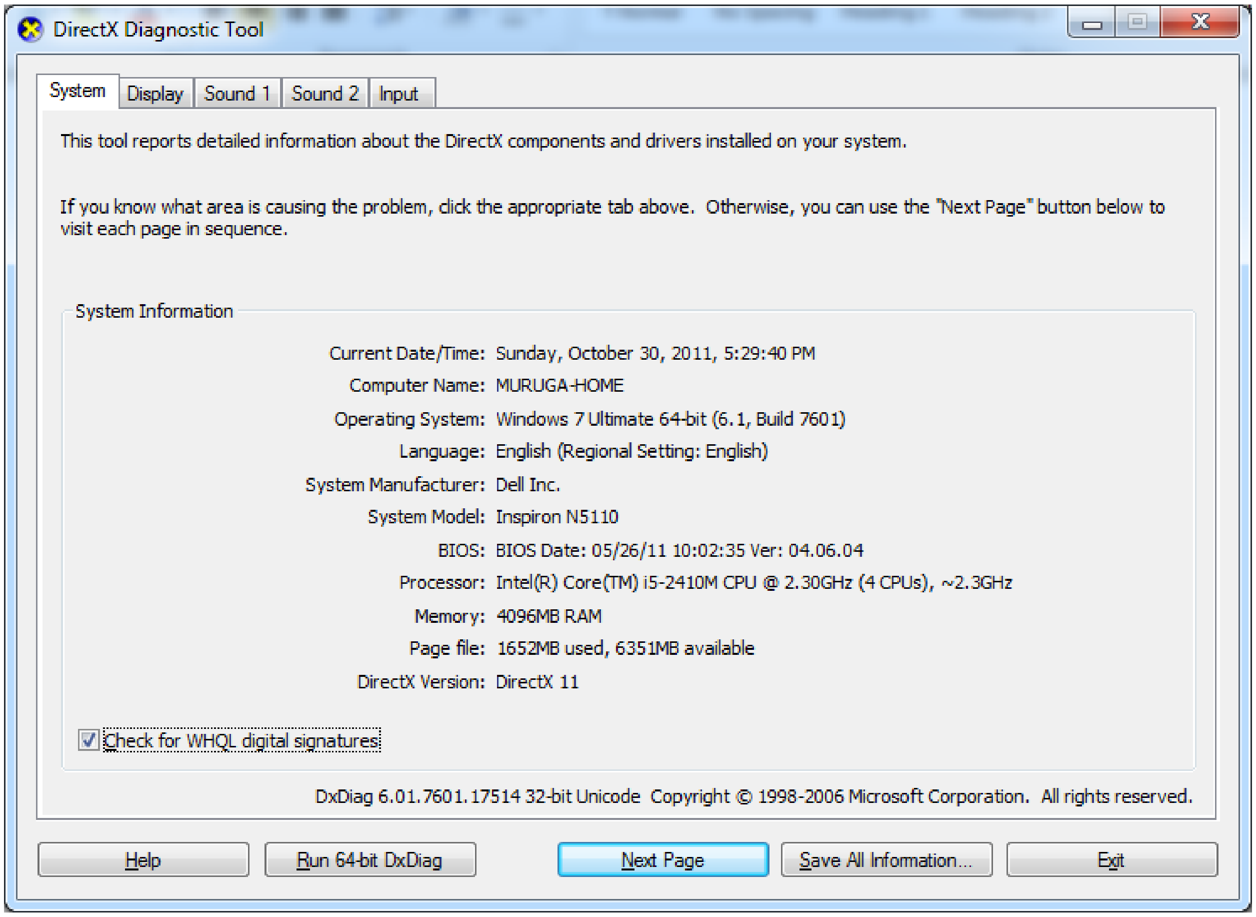
Before we discuss about testing environments, let me list the points I would keep in mind about test environments:

Tip 1: Know the test environment.

Most of the testing competitions specify the test environment to be used by the testers. Do you know the test environment to run your tests? Is the test environment mentioned by the contest owner?

Know which operating system - browser combination you need to test on. If the contest owner has not mentioned the test environment, ask for it. It is better to ask and know before you start.

Operating system, browser, browser version, screen resolution, browser resolution and anti-virus are good starting points to know about the test environment.



This tool reports detailed information about the DirectX components and drivers installed on your system.

If you know what area is causing the problem, click the appropriate tab above. Otherwise, you can use the "Next Page" button below to visit each page in sequence.

System Information

```
   Current Date/Time: Sunday, October 30, 2011, 5:29:40 PM
       Computer Name: MURUGA-HOME
    Operating System: Windows 7 Ultimate 64-bit (6.1, Build 7601)
            Language: English (Regional Setting: English)
 System Manufacturer: Dell Inc.
        System Model: Inspiron N5110
                BIOS: BIOS Date: 05/26/11 10:02:35 Ver: 04.06.04
           Processor: Intel(R) Core(TM) i5-2410M CPU @ 2.30GHz (4 CPUs), ~2.3GHz
              Memory: 4096MB RAM
           Page file: 1652MB used, 6351MB available
     DirectX Version: DirectX 11
```

[✓] Check for WHQL digital signatures

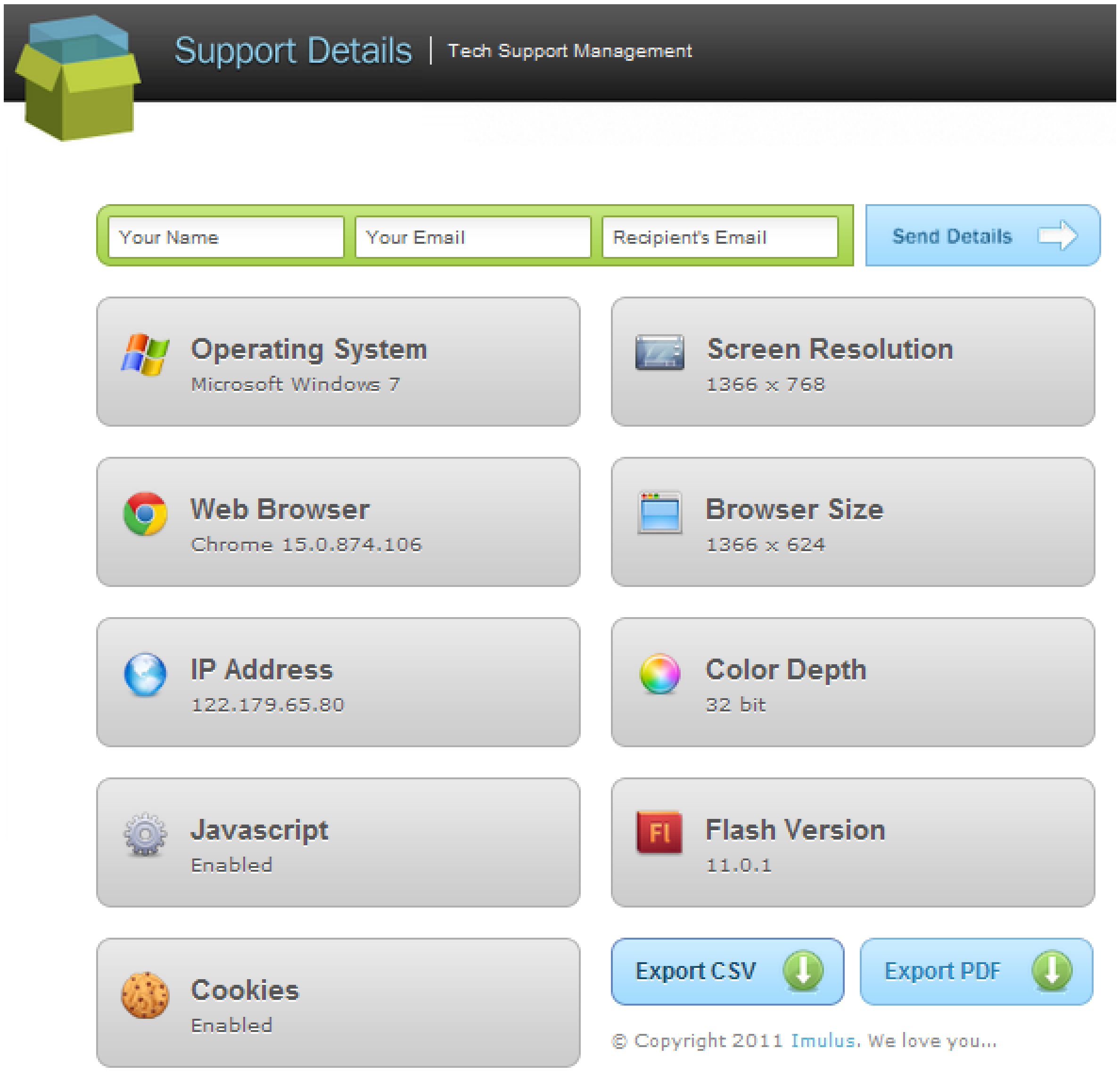
DxDiag 6.01.7601.17514 32-bit Unicode Copyright © 1998-2006 Microsoft Corporation. All rights reserved.

Figure 1: DirectX Diagnostic Tool

Tip 2: Test on the right environment.

Spend most of your time testing on the right environment. The bugs found on the incorrect environment might not be reproducible on the right environment. If you do not have access to the right test environment, test on a different machine which has the right environment. It is very easy to mark a bug invalid when you know that it was found on an incorrect test environment.

Pay special attention to the browser version, screen resolution. One of the easy ways to find the test environment is to browse to www.supportdetails.com.



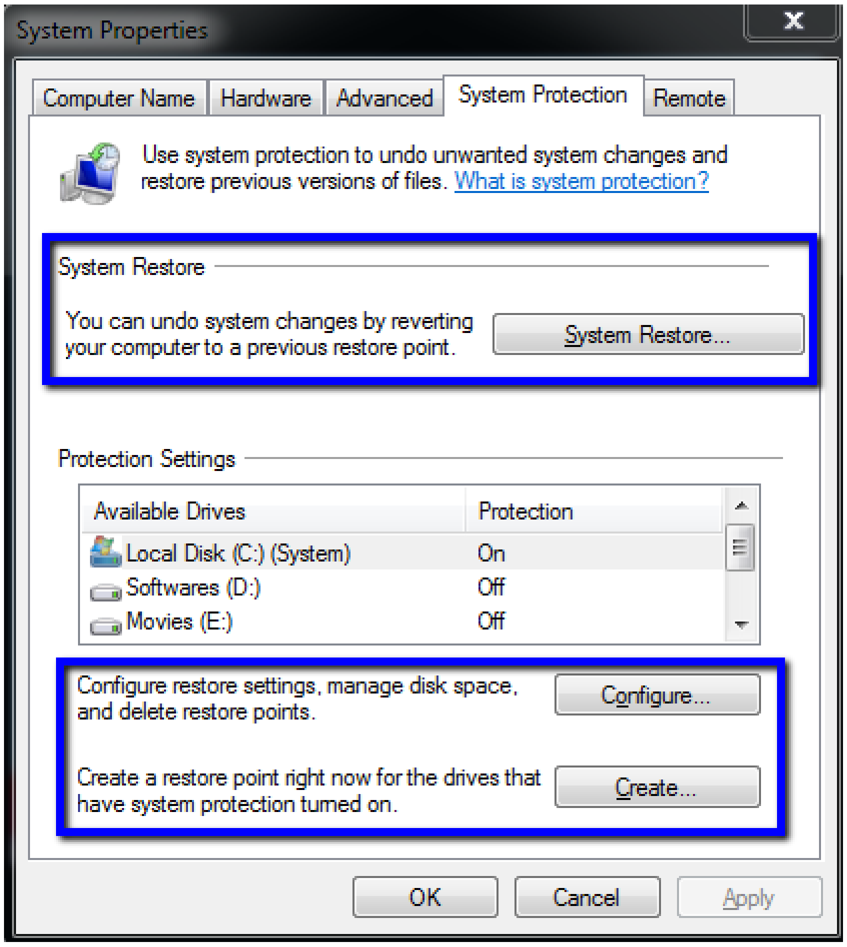
Operating System	Microsoft Windows 7
Screen Resolution	1366 x 768
Web Browser	Chrome 15.0.874.106
Browser Size	1366 x 624
IP Address	122.179.65.80
Color Depth	32 bit
Javascript	Enabled
Flash Version	11.0.1
Cookies	Enabled

© Copyright 2011 Imulus. We love you...

Figure 2: Support Details

Tip 3: Base state of the test environment.

Before starting any tests, ensure that you start from the base state. It helps to restore the base state when you do not want your results to be affected by your earlier tests.



Use system protection to undo unwanted system changes and restore previous versions of files. What is system protection?

System Restore

You can undo system changes by reverting your computer to a previous restore point.

Protection Settings

Available Drives	Protection
Local Disk (C:) (System)	On
Softwares (D:)	Off
Movies (E:)	Off

Configure restore settings, manage disk space, and delete restore points.

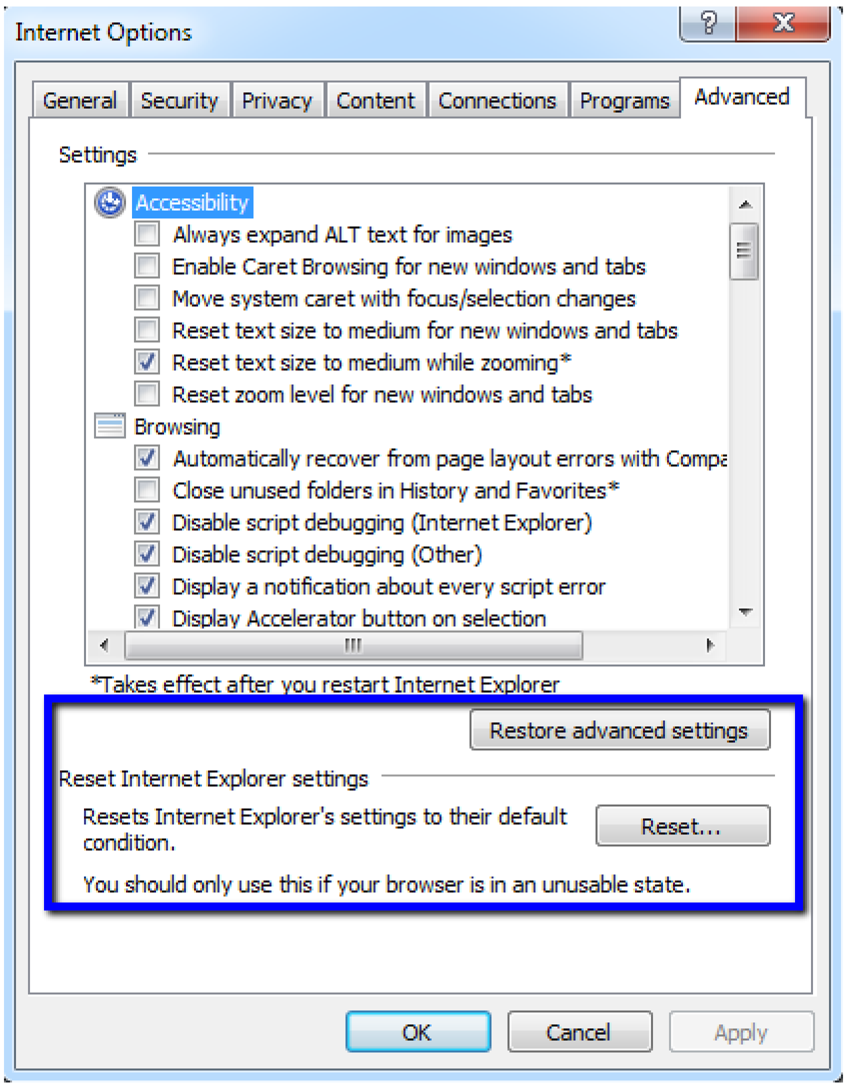
Create a restore point right now for the drives that have system protection turned on.

Figure 3: System Restore

Some testers use Norton Ghost to capture the system state. Suppose you want to save a particular state of the machine, you can take a system backup. Later you can restore it and recreate the exact scenario. It helps in reproducing the hard-to-reproduce bugs.

You can also restore your machine state to the previous known restore point.

In browsers there is an easy option to reset the browser settings to the default condition. Also you can delete the cookies, cache and history to start on a clean state. Some of the advanced settings can also be restored. Some settings take effect only on restarting the machine or browser.



Settings

Accessibility
- [ ] Always expand ALT text for images
- [ ] Enable Caret Browsing for new windows and tabs
- [ ] Move system caret with focus/selection changes
- [ ] Reset text size to medium for new windows and tabs
- [✓] Reset text size to medium while zooming*
- [ ] Reset zoom level for new windows and tabs

Browsing
- [✓] Automatically recover from page layout errors with Compa
- [ ] Close unused folders in History and Favorites*
- [✓] Disable script debugging (Internet Explorer)
- [✓] Disable script debugging (Other)
- [✓] Display a notification about every script error
- [✓] Display Accelerator button on selection

*Takes effect after you restart Internet Explorer

Reset Internet Explorer settings

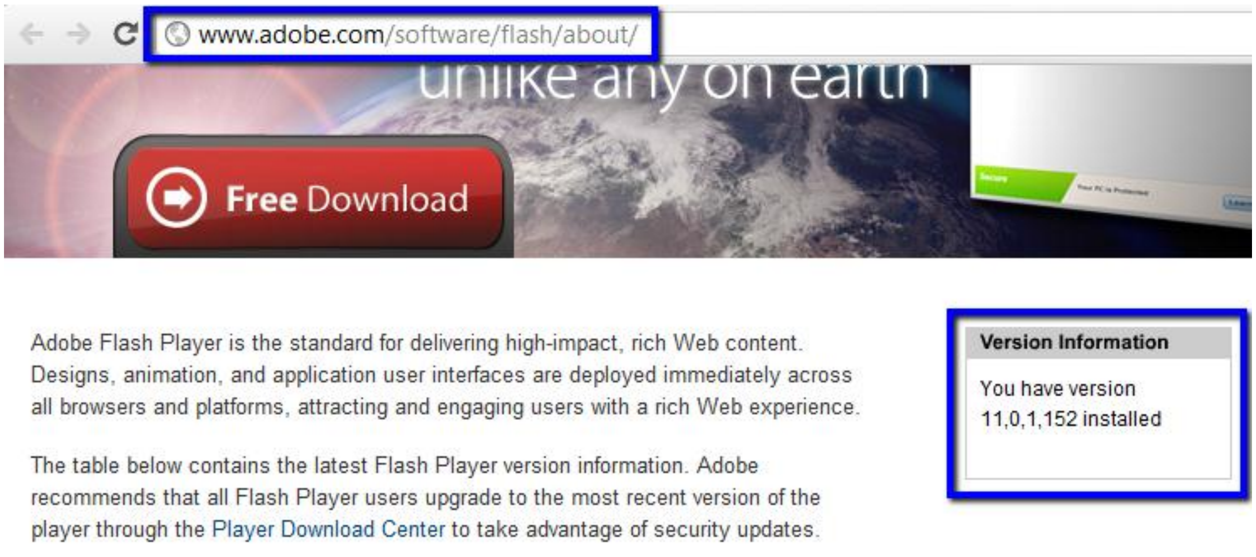
Resets Internet Explorer's settings to their default condition.

You should only use this if your browser is in an unusable state.

Figure 4: Reset IE Settings

Tip 4: Beware of add-ons and third party applications.

Some bugs are dependent on a particular application. Sometimes a different version of flash player or a different Java version might be the reason for a bug. Know the version of the browser add-ons or the third party applications in use. You can check the version of Adobe Flash Player by browsing to http://www.adobe.com/software/flash/about/. You can check the version of Java version by browsing to the java.com/en/download/installed.jsp.



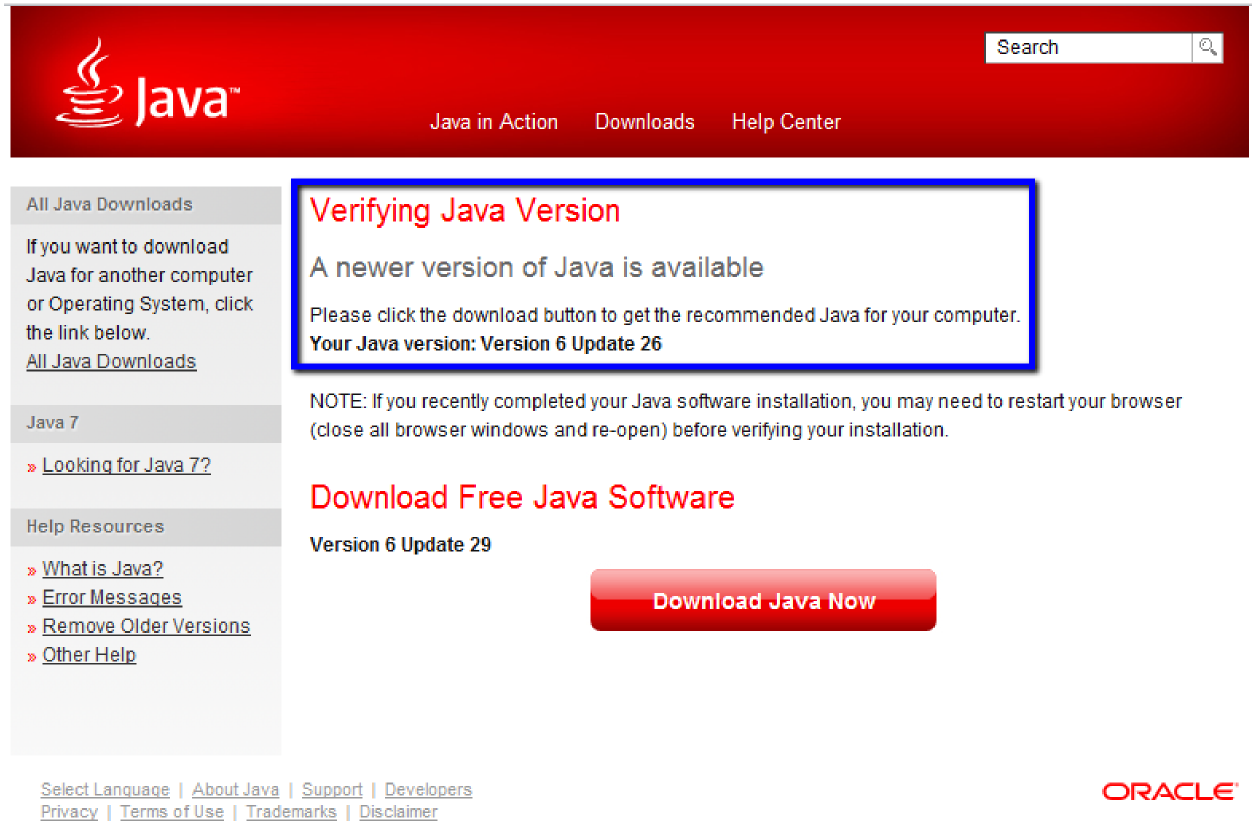
Adobe Flash Player is the standard for delivering high-impact, rich Web content. Designs, animation, and application user interfaces are deployed immediately across all browsers and platforms, attracting and engaging users with a rich Web experience.

The table below contains the latest Flash Player version information. Adobe recommends that all Flash Player users upgrade to the most recent version of the player through the Player Download Center to take advantage of security updates.

Version Information

You have version 11,0,1,152 installed

Figure 5: Adobe Flash Player



All Java Downloads

If you want to download Java for another computer or Operating System, click the link below.
All Java Downloads

Java 7

» Looking for Java 7?

Help Resources

» What is Java?
» Error Messages
» Remove Older Versions
» Other Help

Verifying Java Version

A newer version of Java is available

Please click the download button to get the recommended Java for your computer.
Your Java version: Version 6 Update 26

NOTE: If you recently completed your Java software installation, you may need to restart your browser (close all browser windows and re-open) before verifying your installation.

Download Free Java Software

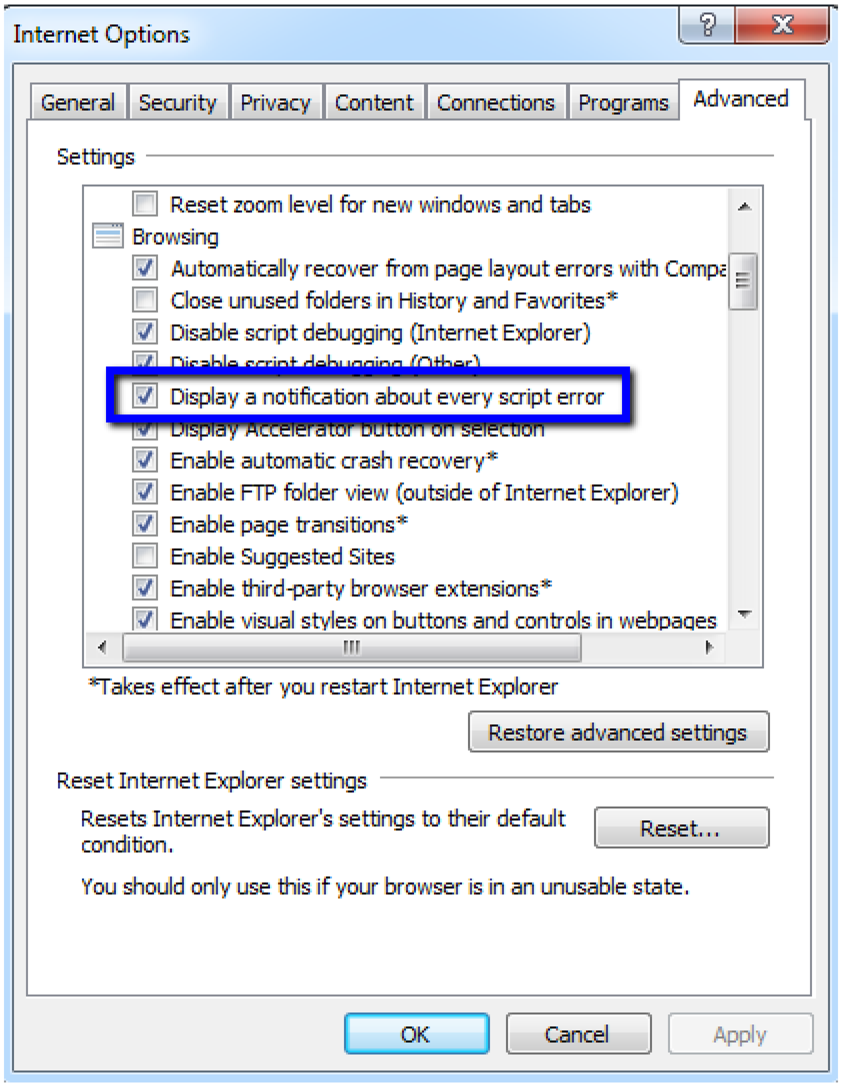
Version 6 Update 29

Select Language | About Java | Support | Developers
Privacy | Terms of Use | Trademarks | Disclaimer

Figure 6: Java Version

Tip 5: Use the test environment to your advantage.

One of the coolest bugs I like to highlight is script errors. It is sometimes easy to miss such bugs by testing on different browsers or by disabling the settings. In Internet Explorer, there is a setting to enable the notification about every script error. Make use of the test environment to find bugs.



Settings

- ☐ Reset zoom level for new windows and tabs
- Browsing
  - ✓ Automatically recover from page layout errors with Compa
  - ☐ Close unused folders in History and Favorites*
  - ✓ Disable script debugging (Internet Explorer)
  - ✓ Disable script debugging (Other)
  - ✓ Display a notification about every script error
  - ✓ Display Accelerator button on selection
  - ✓ Enable automatic crash recovery*
  - ✓ Enable FTP folder view (outside of Internet Explorer)
  - ✓ Enable page transitions*
  - ☐ Enable Suggested Sites
  - ✓ Enable third-party browser extensions*
  - ✓ Enable visual styles on buttons and controls in webpages

*Takes effect after you restart Internet Explorer

Reset Internet Explorer settings

Resets Internet Explorer's settings to their default condition.

You should only use this if your browser is in an unusable state.

Figure 7: Script error notification

To sum up:

- Know the test environment.
- Test on the right environment.
- Preserve the base state.
- Take care to isolate the issues by add-ons and other applications.
- Make use of the test environment settings to find bugs.

Test Planning

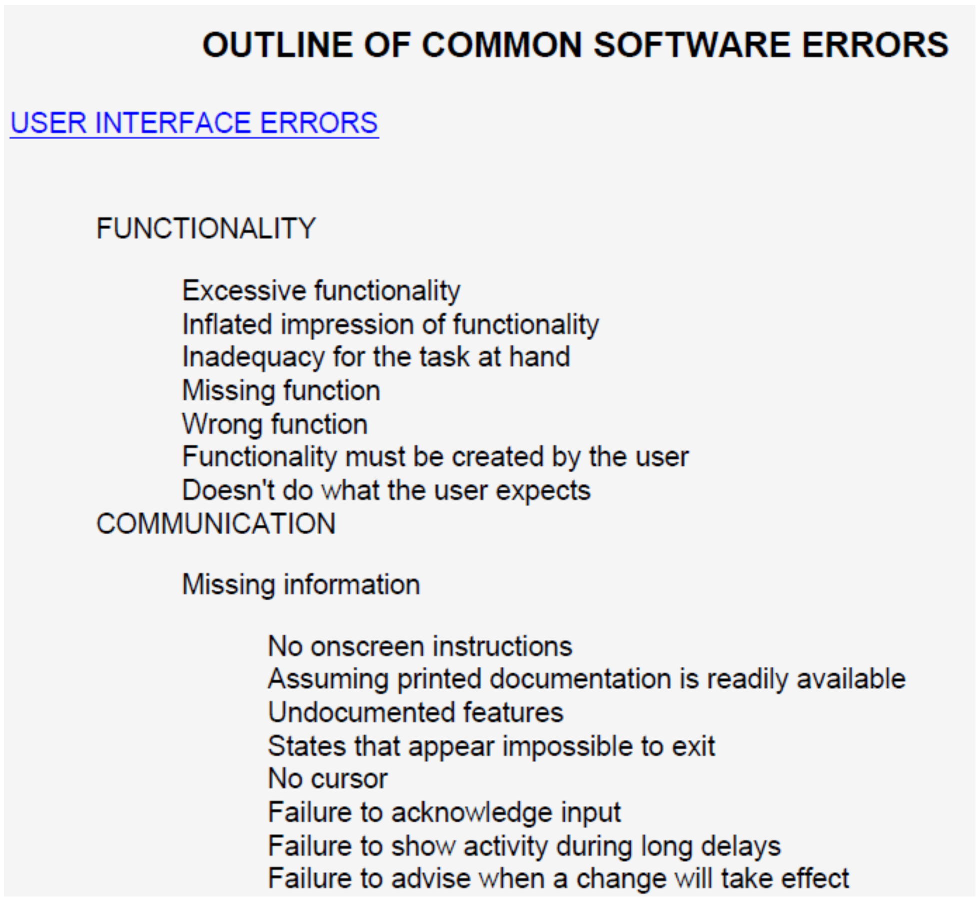
Once you have setup the test environment, the next step is to plan and execute the tests. Having a strong test strategy will help you a lot with the time pressure in testing competitions. In testing competitions that run for few days, every minute is important. The tester who finds the bug first has a good chance compared to others.

Here is my list of tips on test planning:

Tip 6: Suit your test strategy to the competition.

Some testing competitions explicitly state the quality criteria to focus on. If the quality criterion is mentioned, then you need to focus on the stated criteria. Some testers fail to recognize the criteria and find bugs based on other quality criteria. When the stakeholder (contest owner) has clearly mentioned what kind of bugs he needs, focus on the same.

The appendix of Testing Computer Software book is a handy guide to try out different tests. There are close to 300 errors in the list. Browse to http://www.logigear.com/logi_media_dir/Documents/tcs_appA_bugs.pdf and use this list as a starting point.



OUTLINE OF COMMON SOFTWARE ERRORS

USER INTERFACE ERRORS

- FUNCTIONALITY
  - Excessive functionality
  - Inflated impression of functionality
  - Inadequacy for the task at hand
  - Missing function
  - Wrong function
  - Functionality must be created by the user
  - Doesn't do what the user expects
- COMMUNICATION
  - Missing information
    - No onscreen instructions
    - Assuming printed documentation is readily available
    - Undocumented features
    - States that appear impossible to exit
    - No cursor
    - Failure to acknowledge input
    - Failure to show activity during long delays
    - Failure to advise when a change will take effect

Figure 8: Common Software Errors

Tip 7: Play to your strengths.

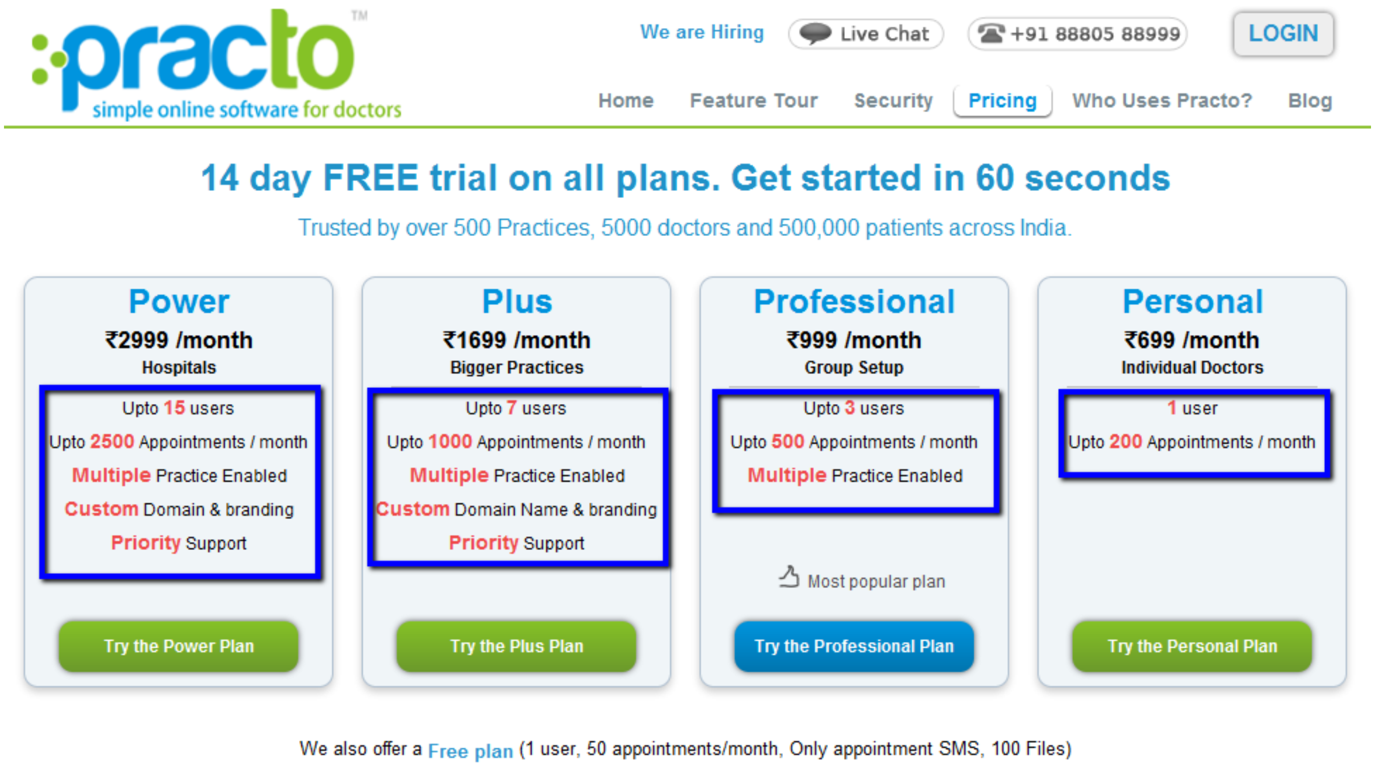
If the contest does not mention anything about the quality criteria, it is time to play to your strengths. Some contests are generic and appreciate any kind of bugs. You might be good in finding functionality bugs or security bugs. Go ahead and report them. Since testing competitions are of short duration, it is better to test based on your existing knowledge than to spend time learning new skills.

Concentrate on the high priority bugs. Many times, a contest is won or lost based on the severity of the bugs than the quantity of bugs.

Tip 8: Focus on unexplored areas.

The payment section is one common area neglected by many testers. When an application involves a payment feature, go ahead and complete the payment scenario. Many testers are stuck at the initial user interface. There are different user privileges based on the payment. Free users might have restricted access to few features while a user with a different plan might have unlimited access to all the features.

Invest in the different plans and get access to features which other testers are unaware of. For example, Practo.com has four different plans. Not many testers would invest in any plan other than the free plan. You can buy any of the plans and test if the claims are true.



14 day FREE trial on all plans. Get started in 60 seconds

Trusted by over 500 Practices, 5000 doctors and 500,000 patients across India.

Power	Plus	Professional	Personal
₹2999 /month	₹1699 /month	₹999 /month	₹699 /month
Hospitals	Bigger Practices	Group Setup	Individual Doctors
Upto 15 users	Upto 7 users	Upto 3 users	1 user
Upto 2500 Appointments / month	Upto 1000 Appointments / month	Upto 500 Appointments / month	Upto 200 Appointments / month
Multiple Practice Enabled	Multiple Practice Enabled	Multiple Practice Enabled	
Custom Domain & branding	Custom Domain Name & branding		
Priority Support	Priority Support		

We also offer a Free plan (1 user, 50 appointments/month, Only appointment SMS, 100 Files)

Figure 9: Different Plans

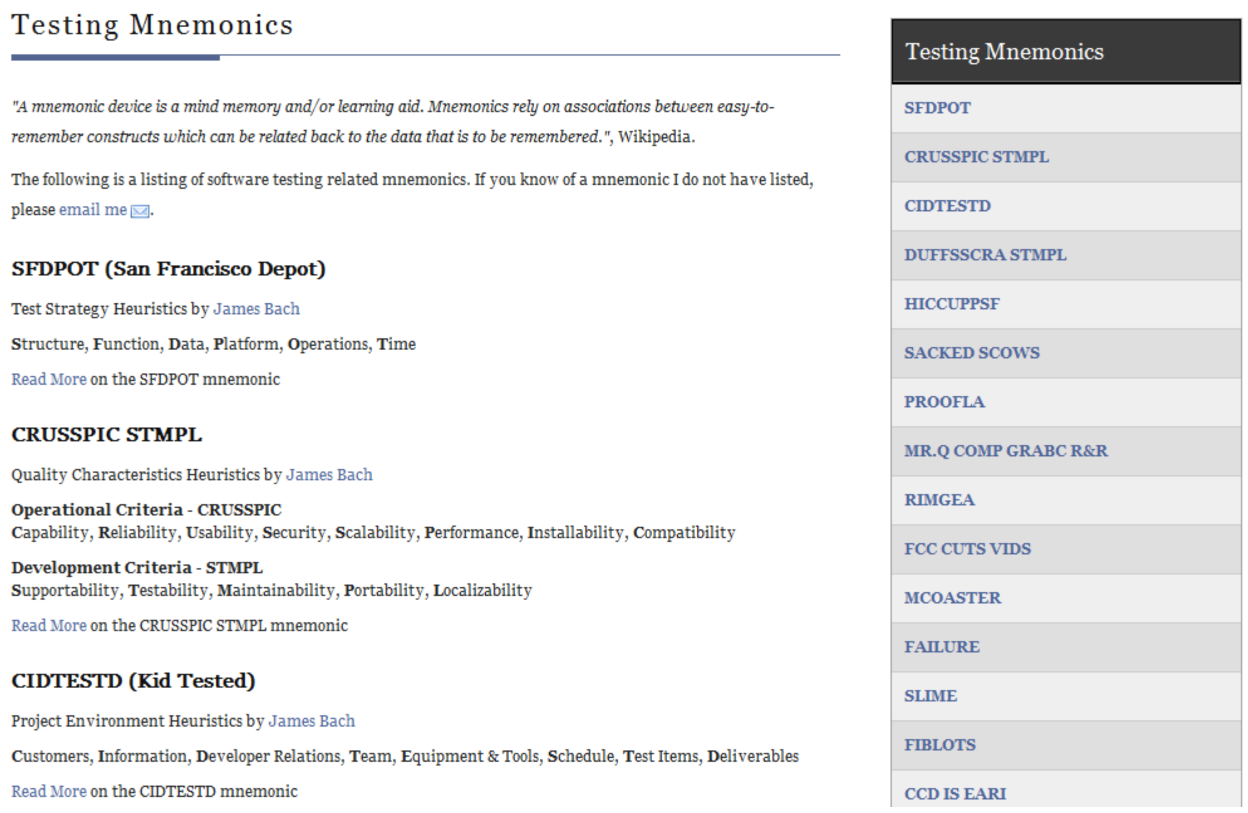
Tip 9: Complete a few user scenarios.

One common mistake testers make is to get stuck on a single screen of the application. Think of a user scenario and try to complete it. Find out if you can complete the scenario or not. Note the problems faced while executing the scenario. Contest owners will be happy with bugs that affect the user workflow. Such bugs will be of higher priority than a bug affecting a single feature.

Tip 10: Concentrate on quick tests.

You need to find the bug before the other tester finds it. A list of ready-to-use mnemonics is a good starting point. Some testers spend a lot of time learning about the application. You need to learn as much valuable information about the application as quickly as possible. Also, the tests you conduct need to be effective and short, yet helps you to find critical bugs. There is no point in spending twenty minutes generating data to find a bug on sorting feature.

Think of cost vs. value. How costly is the test compared to the value it provides? With quick tests, you can discover many bugs in a short time. Have a look at the list compiled by Lynn McKee at www.qualityperspectives.ca/resources_mnemonics.html.



Testing Mnemonics

"A mnemonic device is a mind memory and/or learning aid. Mnemonics rely on associations between easy-to-remember constructs which can be related back to the data that is to be remembered.", Wikipedia.

The following is a listing of software testing related mnemonics. If you know of a mnemonic I do not have listed, please email me.

SFDPOT (San Francisco Depot)

Test Strategy Heuristics by James Bach

Structure, **F**unction, **D**ata, **P**latform, **O**perations, **T**ime

Read More on the SFDPOT mnemonic

CRUSSPIC STMPL

Quality Characteristics Heuristics by James Bach

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

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

Read More on the CRUSSPIC STMPL mnemonic

CIDTESTD (Kid Tested)

Project Environment Heuristics by James Bach

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

Read More on the CIDTESTD mnemonic

Figure 10: Testing Mnemonics - Lynn McKee's Blog

Tip 11: Create your own user credentials.

Usually the contest owners provide you with a set of user credentials for testing purposes. My suggestion is that you do not use them. Create your own set of user credentials. When you use the user credentials provided by the contest owner, you are sharing your tests and test results with other testers. I don’t think that it is a smart idea to share your test ideas. You will never know if other testers are changing the preferences while you are busy testing. It might lead to false positives or missed bugs.

It is worth creating your own set of user credentials thereby testing the user creation, user credentials feature too. New users will be using the new user creation feature first unless the administrator provides them the credentials.

Finally to sum up, here are the main points regarding test planning:

- Understand what is expected from the testers.
- Take advantage of your skillset when there is a choice.
- Test the features which no other tester has tested.
- Think like a user and test for few scenarios.
- Focus on quick tests to find bugs quickly.
- Avoid testing with default user credentials.

Bug Reporting

Good bug reporting is equally as important as finding good bugs. In a testing competition, even the contest owner is under time pressure. The results are expected as soon as possible. Your bug reports should help the contest owner and other stakeholders. It should be easy to understand, easy to validate with all relevant information present in the bug report.

If your bug reports are hard to understand, the contest owner might dismiss your bug report without even trying to reproduce the bugs. Every bug is important in a competition and might play a big role in deciding the final winner. Having done all the hard work of finding the bugs, it is necessary to submit your best bug report.

If I have to submit my bug report I will use the following tips:

Tip 12: Mention all the necessary details.

Nobody likes an incomplete bug report. It irritates the reader more than helping him/her reproduce the issue. A bug report is the voice of a tester. Especially in a testing competition where you cannot meet the contest owner, you have to make sure the bug report conveys all the information. Let me describe a few important components of a bug report.

- A summary or description of the bug.

The first impression of a bug is conveyed by the summary or description of the bug. If someone asks you what is the issue, the summary must be able to answer the question in a single sentence. Very few have the patience to read a long summary of the bug.

- Steps to reproduce.

Mention the steps to reproduce the bug. Sometimes, the person who validates the bug might be a non-technical person. He/she might not be aware of the details of the application. The steps to reproduce must fill this gap. Write easy to understand steps. Do not miss any step. Remove any unnecessary steps.

For example, on a website if I want the user to view the ‘Invoice History’ page, I can choose one of the following steps:

1. Login > My Profile > Payment History > Invoice History

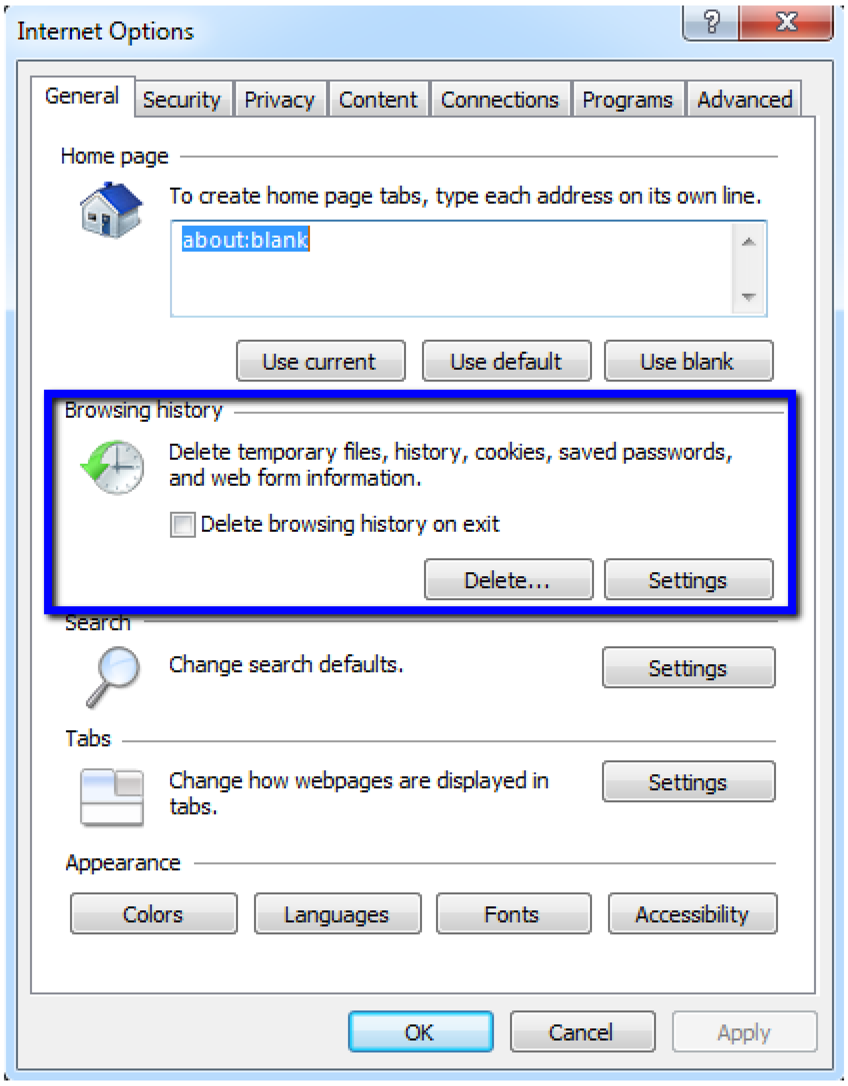
2. Login > Browse to www.site.com/InvoiceHistory.aspx

In many bug reports, I see testers mentioning the first step. There is no need to move through the Profile and Payment pages. As long as the user is logged in, if he/she clicks on the Invoice History page link, the page is displayed. If the user cannot browse to the final page without the intermediate steps, then it makes sense to mention the intermediate steps.

- Start from the base state.

The person validating your bugs does not validate the bugs on your machine. Before reporting the bug, try to reproduce the bug from the base state. The current buggy state of the application might be because of your previous tests. If the contest owner follows the steps to reproduce, he/she might not be able to reproduce the bug. In your bug report, mention the base state. Start your test from the base state.

Sometimes a base state might be as simple as clearing the cookies and cache.



Home page

To create home page tabs, type each address on its own line.

about:blank

Browsing history

Delete temporary files, history, cookies, saved passwords, and web form information.

☐ Delete browsing history on exit

Search

Change search defaults.

Tabs

Change how webpages are displayed in tabs.

Appearance

Figure 11: Clear browsing history

- Attach a screenshot or video.

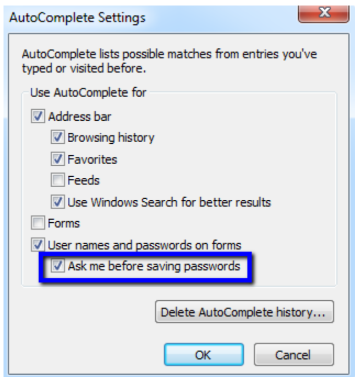
A screenshot of the issue is very useful for the contest owner. There might be confusing terminology and a screenshot highlights the issue immediately. If the screenshot is not appropriate or does not convey all the information, attach a video. If the issue needs the user to navigate through multiple screens, capture a video and attach it.

- Mention the number of attachments.

It is a good practice to mention the number of attachments. There is a possibility that no one notices the attachment. It might be hard to recreate the bug without the attachment. Mentioning the attached file names and number of attachments lets the user know about the attachments. If you mention the attachments but fail to attach, the contest owner might comment on your bug. You might later attach the files.

Tip 13: Mention special settings if any.

Any special setting on your machine might be the cause of the bug. Make sure that you do isolate any such special settings. If they are necessary to recreate the bug, mention the setting. The contest owner will not like to spend a lot of time trying to recreate an issue on an incorrect setting. A few special settings include:



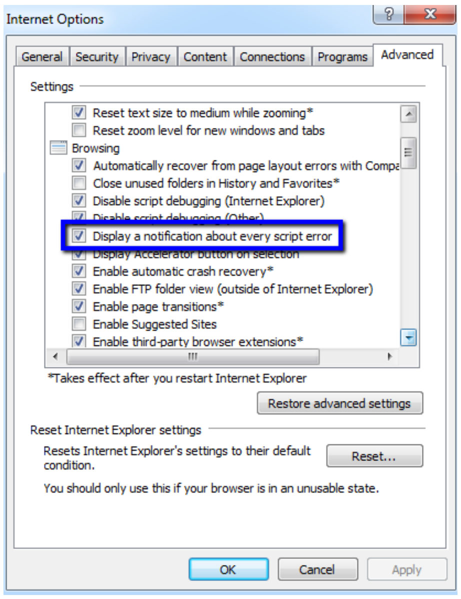
AutoComplete Settings

AutoComplete lists possible matches from entries you've typed or visited before.

Use AutoComplete for

- ✓ Address bar
  - ✓ Browsing history
  - ✓ Favorites
  - ☐ Feeds
  - ✓ Use Windows Search for better results
- ☐ Forms
- ✓ User names and passwords on forms
  - ✓ Ask me before saving passwords

Figure 12: AutoComplete Settings



Internet Options

Settings

- ✓ Reset text size to medium while zooming*
- ☐ Reset zoom level for new windows and tabs
- Browsing
  - ✓ Automatically recover from page layout errors with Compa
  - ☐ Close unused folders in History and Favorites*
  - ✓ Disable script debugging (Internet Explorer)
  - ✓ Disable script debugging (Other)
  - ✓ Display a notification about every script error
  - ✓ Display Accelerator button on selection
  - ✓ Enable automatic crash recovery*
  - ✓ Enable FTP folder view (outside of Internet Explorer)
  - ✓ Enable page transitions*
  - ☐ Enable Suggested Sites
  - ✓ Enable third-party browser extensions*

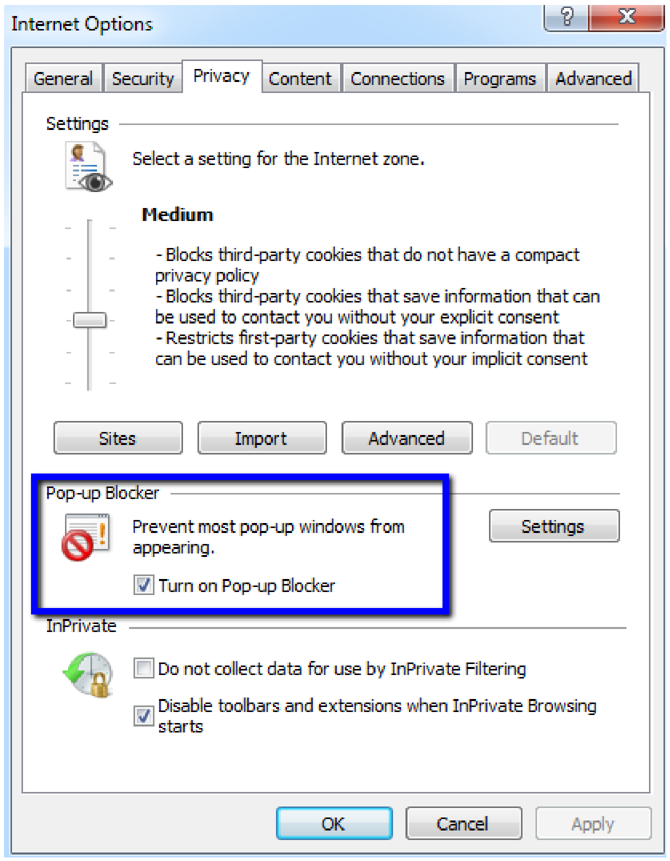
*Takes effect after you restart Internet Explorer

Reset Internet Explorer settings

Resets Internet Explorer's settings to their default condition.

You should only use this if your browser is in an unusable state.

Figure 13: Notification



Settings

Select a setting for the Internet zone.

Medium

- Blocks third-party cookies that do not have a compact privacy policy
- Blocks third-party cookies that save information that can be used to contact you without your explicit consent
- Restricts first-party cookies that save information that can be used to contact you without your implicit consent

Pop-up Blocker

Prevent most pop-up windows from appearing.

☑ Turn on Pop-up Blocker

InPrivate

☐ Do not collect data for use by InPrivate Filtering

☑ Disable toolbars and extensions when InPrivate Browsing starts

Figure 14: Pop-up Blocker

Tip 14: Use the 'Edit bug' feature.

In testing competitions, the tester who logs the bug first gains the points. Even if your bug report is better than the other tester's report, whoever logs the bug first gets the point. Here is one tip I regularly use. If you can edit a bug, log a bug with just one line summary and description. Get your bug registered in the bug database before other testers log the same issue.

Once you have logged the issue, you can edit the bug and complete the bug report later. By following this tip, we can save time and log the bug before other testers log it.

Tip 15: Highlight how the user is affected.

As a tester when you find the bug, you understand the impact of the bug on the user. Use this knowledge to your advantage. Highlight it in your bug report. Let the contest owner know the impact of the bug. Mention how the bug will affect the user and how the contest owner might lose business or reputation. Business owners talk in the language of money and as a tester, you need to highlight this potential loss if a particular bug is not fixed. A simple point highlighting the number of users who will encounter the issue or the cost of support calls will let the contest owners know the importance of the bug. By highlighting the loss, you can increase the chance of getting your bug fixed.

Tip 16: Have the bug summary stand out.

A short and catchy summary is important to make your bug stand out among hundreds of other bugs. The first impression of any bug is its bug summary. A bug summary must be short but at the same time, have enough information. It can start with the feature name where the issue occurs. Try to provide as much information in the first few words of the summary. The longer the summary, the chances of people reading the entire bug summary reduces drastically.

Do not mention all the entire steps in the bug summary. 'Steps to reproduce' needs to be mentioned in the 'Steps' section.

Example:

Bad Summary: After launching the application and closing it, the application cannot be launched and machine needs to be restarted.

Good Summary: Application fails to launch after first launch.

If we read the first summary, the first ten words do not say anything about the issue. Compare it with the next summary. The entire issue is highlighted in less than eight words.

There is no need to mention about the machine start in the summary. It can be highlighted in the 'Observed Behavior' section.

Tip 17: Assign the appropriate severity.

As a tester, you might need to assign the severity to the bug. Points are assigned to every valid bug based on the severity. For example, showstopper bugs can fetch you ten points, high, medium and low bugs can fetch you six, four and two points respectively. Many testers assign a severity as high as possible to get more points.

If the tester assigns an incorrect severity, the contest owner has to change the severity. It is additional work for the contest owner. When a tester gives additional work to the contest owner, it gives a bad impression about the tester. Even if the tester logs a valid bug, the contest owner does not give much attention. The credibility of the tester is reduced. For a short term gain, the tester increases the severity. But in the long term, the tester does more harm to himself.

I follow the following guidelines to assign the four severity options:

Showstopper: Any script errors, blocked features, high loss to customer.

High: No workaround for a particular feature, many steps to be repeated.

Medium: Workaround available, user can continue to use the feature.

Low: Cosmetic issues, spelling mistakes, GUI issues, no major loss if bug is not fixed.

Tip 18: Have a one-liner bug report.

In the previous tips I highlighted that the bug reports need to be complete. I suggested you to provide as much information as possible for every bug report. In this tip, I ask you to provide as less information as possible in the bug report. When the deadline is nearing, do not spend much time providing many details to your bug report. Provide just enough information. Maybe a one line summary and a screenshot will be enough.

As these bugs will be validated at the end stages, even the contest owner will be thinking of wrapping up this exercise soon. Go ahead and try your luck. The last few bugs might be the difference between the first and second prize.

Tip 19: Initial bug reports to final reports.

First impressions matter a lot. When the contest owner validates the bugs, he/she will get a fair idea of which tester logs what kind of bugs. One can recognize the type of bugs logged by a particular tester. So, put in more effort initially. Spend time writing as detailed bug reports as possible when you start the contest. Let the contest owner get a good impression about you and your bug reports. Later as the deadline approaches, you can spend more time on finding bugs than writing detailed bug reports.

There are testers who are consistent throughout the contest. They spend equal amount of time finding bugs and reporting bugs right from the start to end of the contest. Get ahead of such testers by spending more time finding bugs. At this stage of the contest, quantity of bugs matter than quality of bug reports. It also depends on how well you have drafted your initial bug reports.

In a nutshell, take care of following points:

- Mention all the necessary details for your bug report.
- Mention the special settings on your machine.
- Use the ‘Edit bug’ feature.
- Highlight the bug’s impact on the customer.
- Let your bug summary attract attention.
- Assign the right severity.
- Make use of one-liner bug reports.
- Concentrate less on bug reports during the last phase of contest.

Learn from other bugs

Tip 20: Note the quality criteria.

What quality criterion is being used by other testers? While you are focused on a single criterion, are others testing based on different criteria? Are you missing out on a lot of bugs or on important bugs? By going through the list of bugs, you can understand where the majority of bugs are discovered till now. You might learn about a new test idea. You can build on the test idea or use it on other projects or contests.

Tip 21: Features tested and not tested.

One of the advantages of many testers testing the same application is that you can achieve a lot of coverage quickly compared to a few testers testing for the same duration. As a tester, you cannot test every feature in a given span of time. Make use of the bugs logged so far to know which features are tested. If you find a lot of bugs logged for a particular feature, it could mean that most of the bugs have been already identified by the testers. Will you continue to test the feature or focus on a different feature?

You will also get to know which features of the application are not tested or no bugs have been logged. When you read the bugs logged, it gives you a fair idea of what kind of tests revealed the bugs. Have the other testers missed obvious tests?

Tip 22: Note the techniques used.

Every tester has a unique style of testing. A cost effective way to learn from other testers is to go through the bugs logged by them. Why did you miss the bug? Was it because you did not think of the test idea or you did not execute the test idea? Do you know the technique used to discover the bug? In the book - Lessons Learned in Software Testing by Cem Kaner, James Bach and Bret Pettichord, the five dimensions - testers, coverage, potential problems, activities and evaluation is described. Which dimension will you focus on? Which dimension other testers are using?

For testing contests, there will be a variety of techniques used. Why are you not using a particular technique? Are you aware of why you are using or not using a technique?

Tip 23: Build on others' bugs.

There are many testers who seem to be in a hurry when testing for testing contests. As soon as they find an error, they report it without much investigation. They think that logging bugs as soon as they find is much more important than investigating or searching for a severe issue. If we go through some of the bugs logged, we will discover that the issues are just the symptoms of a bigger problem. There might be more sever issues hiding behind these small bugs.

Instead of logging three to four cosmetic issues, you can find two to three high or even showstopper bugs. For the same amount of time invested, you can find more critical bugs. Use others' bugs as a starting point. Build on those bugs.

It is time to sum up the tips on learning from others' bugs:

- Are you missing out on any quality criteria?
- Know which features have been already tested by other testers.
- Can you learn any new techniques from this contest?
- Build on bugs logged by other testers.

Tools

As a tester, you need to be always prepared to attack any application. In testing contests, there is constant time pressure. Tools can help any tester to a great extent. You need to know how to use those tools effectively. Also, you must be able to use them under extreme time pressure. One way to learn a tool well is to keep using it and practicing it when there is no time pressure. Make mistakes, learn from them and do not repeat the mistakes. In the following tips, I describe the tools I like and use the most during testing contests.

Tip 24: Good Screen capture and video recording tools.

As we saw in the Bug reporting section, attachments in the form of screen capture or a video file adds a lot of value to your bug. Try out different tools and select the one which suits your needs the most.

Before selecting a good screen capture tool, think of these points:

- Quick to respond to a key click or hotkey.

You do not want to miss out on a bug due to the time the tool takes to launch/capture the screen.

- Customizable hotkey and filename.

Some tools have pre-defined key combination for taking a screenshot. Can you change it to a combination which is easy to remember and press when necessary? Do you have to enter a file name for every capture?

- Captures mouse position and tooltips.

Does it capture the mouse pointer position? Many tools do not capture the tooltips. Ensure that the tool you select captures both mouse pointer and tooltips. Maybe you need to change the setting in preferences section.

- Autosave to a predefined folder.

Every time you capture a screenshot, if a confirm save dialog pops up, you lose valuable time. Instead look for a tool which automatically saves all your screenshots to a pre-defined folder. Remember to check if you can customize the folder.

- Editing capability

What is the capability of this tool in terms of editing the screenshot? Can you highlight, crop, scale, color and add text to this screenshot?

For video recording tools, think in terms of

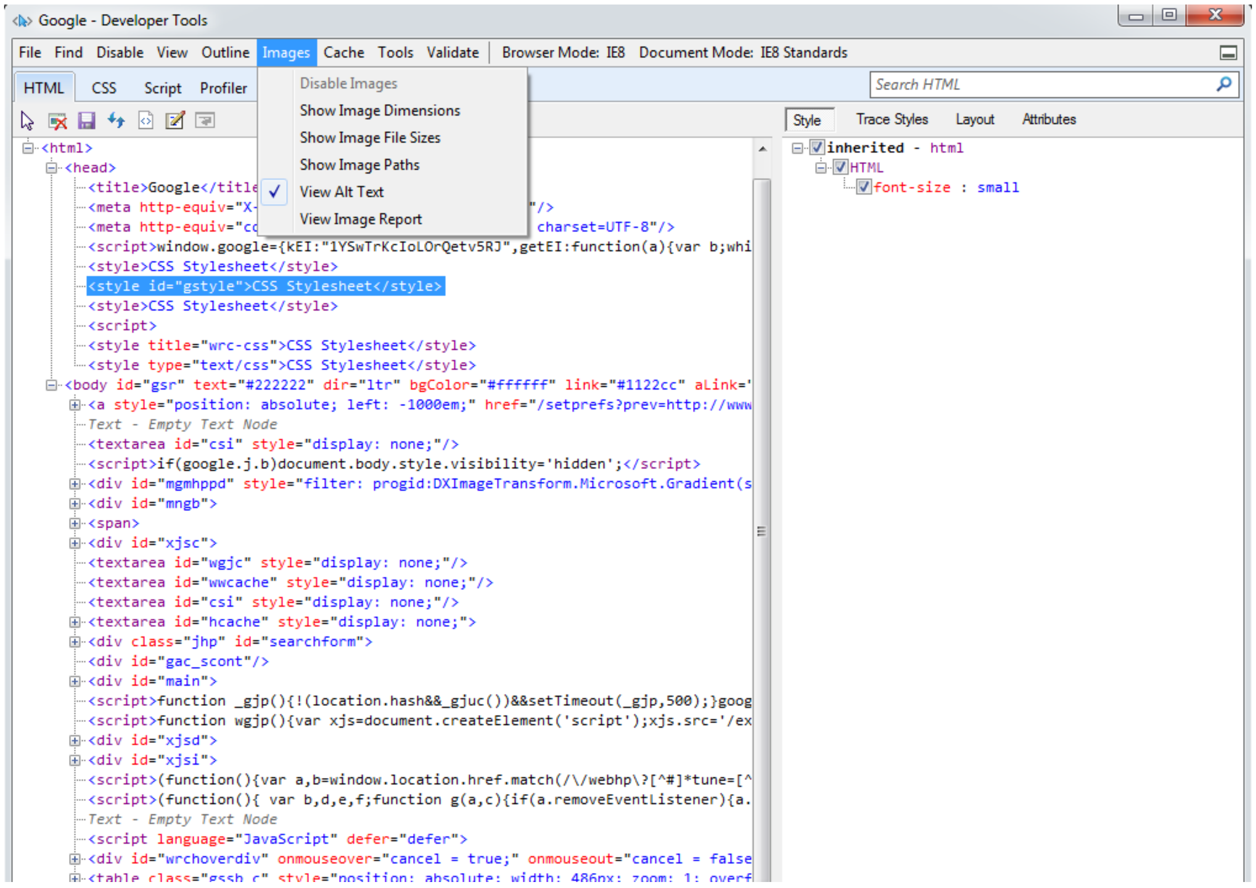
- Size
- Output format

- Quality options
- Settings
- Audio and video
- Highlighting options

I recommend Jing http://www.techsmith.com/download/jing/ and FastStone Capture http://faststone.org/FSCaptureDetail.htm.

Tip 25: Browser add-ons and developer tools.

Some tests can be performed better if you enable the developer tools. Using these tools, you can have a preview of the code. New test ideas might pop up or you might get proof for the existing bugs.



```
<html>
  <head>
    <title>Google</title
    <meta http-equiv="X-                                  "/>
    <meta http-equiv="co                                  charset=UTF-8"/>
    <script>window.google={kEI:"1YSwTrKcIoLOrQetv5RJ",getEI:function(a){var b;whi
    <style>CSS Stylesheet</style>
    <style id="gstyle">CSS Stylesheet</style>
    <style>CSS Stylesheet</style>
    <script>
    <style title="wrc-css">CSS Stylesheet</style>
    <style type="text/css">CSS Stylesheet</style>
  <body id="gsr" text="#222222" dir="ltr" bgColor="#ffffff" link="#1122cc" aLink='
    <a style="position: absolute; left: -1000em;" href="/setprefs?prev=http://www
    Text - Empty Text Node
    <textarea id="csi" style="display: none;"/>
    <script>if(google.j.b)document.body.style.visibility='hidden';</script>
    <div id="mgmhppd" style="filter: progid:DXImageTransform.Microsoft.Gradient(s
    <div id="mngb">
    <span>
    <div id="xjsc">
    <textarea id="wgjc" style="display: none;"/>
    <textarea id="wwcache" style="display: none;"/>
    <textarea id="csi" style="display: none;"/>
    <textarea id="hcache" style="display: none;">
    <div class="jhp" id="searchform">
    <div id="gac_scont"/>
    <div id="main">
    <script>function _gjp(){!(location.hash&&_gjuc())&&setTimeout(_gjp,500);}goog
    <script>function wgjp(){var xjs=document.createElement('script');xjs.src='/ex
    <div id="xjsd">
    <div id="xjsi">
    <script>(function(){var a,b=window.location.href.match(/\/webhp\?[^#]*tune=[^
    <script>(function(){ var b,d,e,f;function g(a,c){if(a.removeEventListener){a.
    Text - Empty Text Node
    <script language="JavaScript" defer="defer">
    <div id="wrchoverdiv" onmouseover="cancel = true;" onmouseout="cancel = false
    <table class="gssh_c" style="position: absolute; width: 486px; zoom: 1; overf
```

```
inherited - html
  HTML
    font-size : small
```

Figure 15: Developer Tools

Some browser add-ons are especially useful. A good list of browser add-ons is mentioned on Santhosh Tuppad's blog: http://tuppad.com/blog/2011/10/26/how-to-test-better-with-add-ons/.

There are two mindmaps highlighting only browser add-ons. They are published on Moolya's blog.

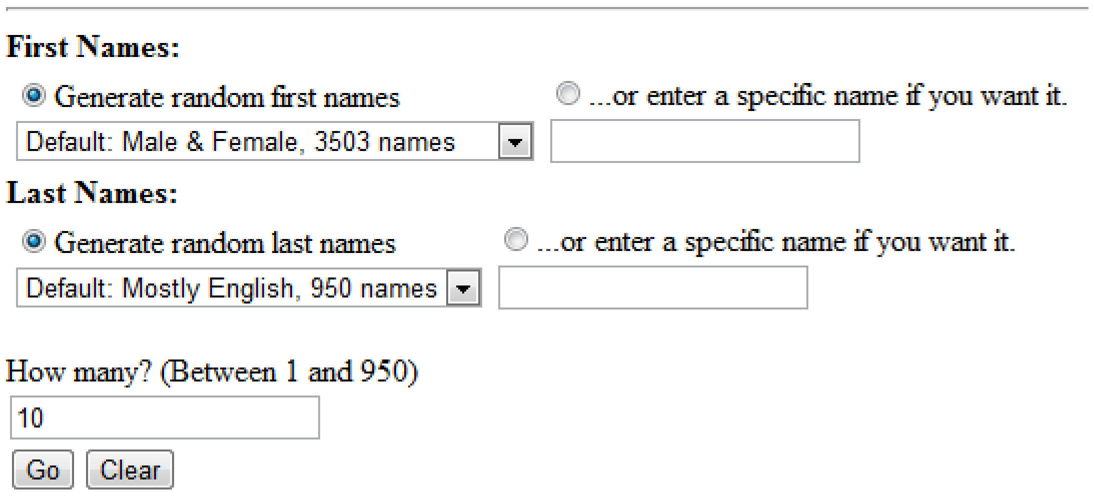
http://moolya.com/blog/2011/03/04/addon-mindmap-for-testers-from-moolya/.

Use them and hunt for more add-ons which will support your testing.

Tip 26: Test data generators.

Some features demand a lot of test data. For example, imagine testing an application that needs a minimum of fifty users. What names do you give to the fifty users? How about generating the names in a matter of few clicks?

http://www.xtra-rant.com/gennames/ generates random first name and last name. You can choose to have one of them constant.



First Names:

(●) Generate random first names ( ) ...or enter a specific name if you want it.

Default: Male & Female, 3503 names

Last Names:

(●) Generate random last names ( ) ...or enter a specific name if you want it.

Default: Mostly English, 950 names

How many? (Between 1 and 950)

10

Figure 16: Name Generator

I found http://www.markrichman.com/2007/09/26/generating-random-names-as-test-data/ when I was searching for name generators. He has 60,000+ names listed in an excel sheet. You can download the excel sheet and use the data for your testing.

One more tool I regularly use is **Perlclip** by James Bach and Danny Faught. It can be downloaded from http://www.satisfice.com/tools.shtml. Perlclip helps me a lot especially when I have to test the limits of a field or generate a lot of data.

I love one more tool and I happily recommend it. It's **Hexawise** http://hexawise.com/**.** Many testers face the confusion of which combinations to test, which combinations to ignore and want to have maximum coverage. Hexawise by Justin Hunter will help you get started.

Figure 17: Hexawise

Tip 27: Credit card details.

For testing payment feature, you might need credit cards. You can use your real and valid credit cards, spend real money and test the feature. Or you can use fake credit cards, good enough to test the features. Which one do you prefer? On Google, I found a site which had a list of fake credit card numbers. To my surprise, it worked and I could test the payment feature of few applications. I will not give the website of the credit cards here. You can easily Google it.

Tip 28: Use Dropbox and drop your worries.

When testing, you might have to use different computers or machines on different networks. You might have your notes on one machine while files on another. What do you do? Use pen drives, hard disks and transfer them, or use dropbox on both machines? I wrote a blog post on how dropbox is useful in my testing here: http://enjoytesting.blogspot.com/2011/07/dropbox-i-love-you.html. Dropbox plays a vital role in my testing. Even if there is a power cut, I know that I can access my files from Dropbox.

UM... YOU DO KNOW THAT YOU CAN USE DROPBOX STRAIGHT FROM YOUR PHONE, RIGHT?
OH... REALLY...?

Figure 18: Dropbox

Recollecting the tips on tools:

- Have your screen capture and video recorders ready.
- Make use of browser add-ons and developer tools.
- Test data generators, Perlclip and Hexawise save a lot of testing time.
- Use fake credit cards to test payment features.
- Use Dropbox to save your files on web and across computers.

Documentation

Testing contests are not like your regular testing projects. Your stakeholders might not expect lots of documentation other than the bug reports. As a tester, you might not have access to many documents too. What would be your approach in such cases? How do you test? Do you write test cases or do you save your testing results? Is it worth spending some time reading the documentation provided by the contest owner?

Tip 29: Read the user guides and other documents.

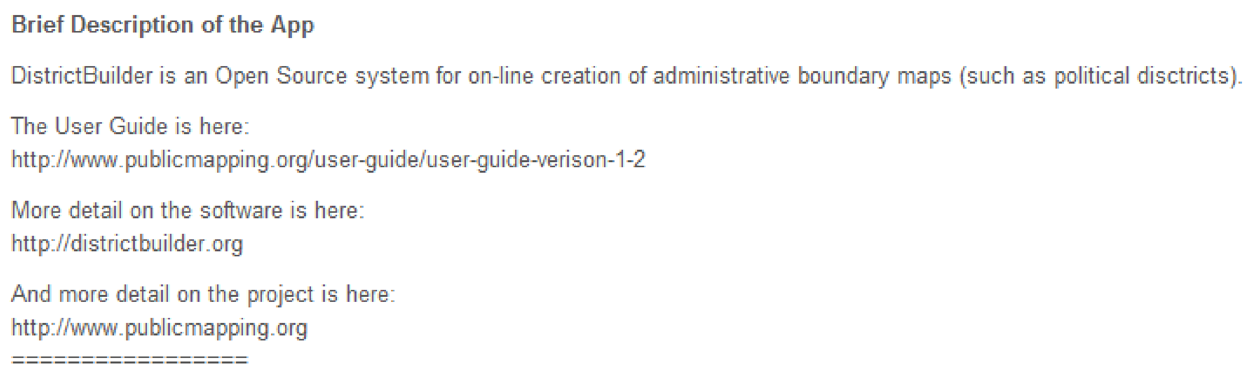
One common mistake I have noticed in many testing contests is that testers log bugs without understanding the feature. Why did the testers not understand the feature - they did not go through the documents. Do not make that mistake. Read the user guides and other documents. Use them to

a) Generate test ideas.

b) Assess your coverage in terms of features.

c) Understand the functionality.

If you feel that it is a huge document, skim the document, read quickly and make notes. There is no need to read the full document.



Brief Description of the App

DistrictBuilder is an Open Source system for on-line creation of administrative boundary maps (such as political disctricts).

The User Guide is here:
http://www.publicmapping.org/user-guide/user-guide-verison-1-2

More detail on the software is here:
http://districtbuilder.org

And more detail on the project is here:
http://www.publicmapping.org
=================

Figure 19: Documentation

Tip 30: Concentrate on few sections.

For applications with huge user guides and multiple documents, use your knowledge gained from reading other bug reports. Know which feature is covered the least. Read only few sections of the user guide. Instead of being a master of all the features, own only a few features. Read about them and test them thoroughly.

By following this approach, the entire user guide is reduced to few pages. This is easy to concentrate on. Some testers fall into the trap of reading the user guide from first page to last page. A lot of time is wasted. You can save a lot of time. But make sure you know everything about the limited feature set.

Tip 31: Release notes and bug fixes.

As soon as a product is released for testing, visit the website where the live product is released. Most of the contest owners provide a demo site, yet to be released product than a live site. The live site might have release notes for the latest download version. By reading the release notes, you will know what the new additions are - maybe a new OS/browser is supported, a new feature is added/removed or new GUI changes. This gives a fair idea of where most of the code changes have gone in. It is valuable to focus your testing time on these new features.

The bug fixes document lists the bugs which were fixed in the release. By going through this document, we can learn about

- When bugs were fixed.
- Which bugs are important to the stakeholders.
- What bugs the customers face.
- Which features had the most bug fixes.

Use this knowledge to your advantage. Also, remember that this count of bug fixes acts as a rough guide. There might be more important bugs which are not fixed or other features with more bugs. Use this knowledge as a heuristic and remember that anything, including heuristics, *may* fail.

Tip 32: Documentation on your part.

As a tester, what kind of documents will you prepare for the testing contests? One mandatory piece of this documentation is the bug reports which will be filed in the bug database. Apart from the bug reports, do you need to write test cases, checklists, or mind maps? I am not a supporter of test cases. You know the least about the application during the start of the contest and writing test cases takes up a lot of valuable time.

You may prepare a mind map of the application with the features, test ideas and any other information. A mindmap gives a quick visual overview of the application. You can focus on a part of the map to start with or you can have a map with your testing strategy. This is one of the mindmaps I used for a contest:

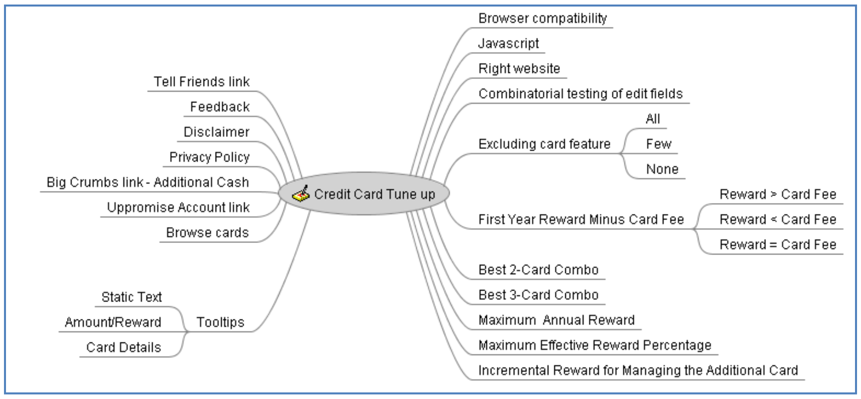
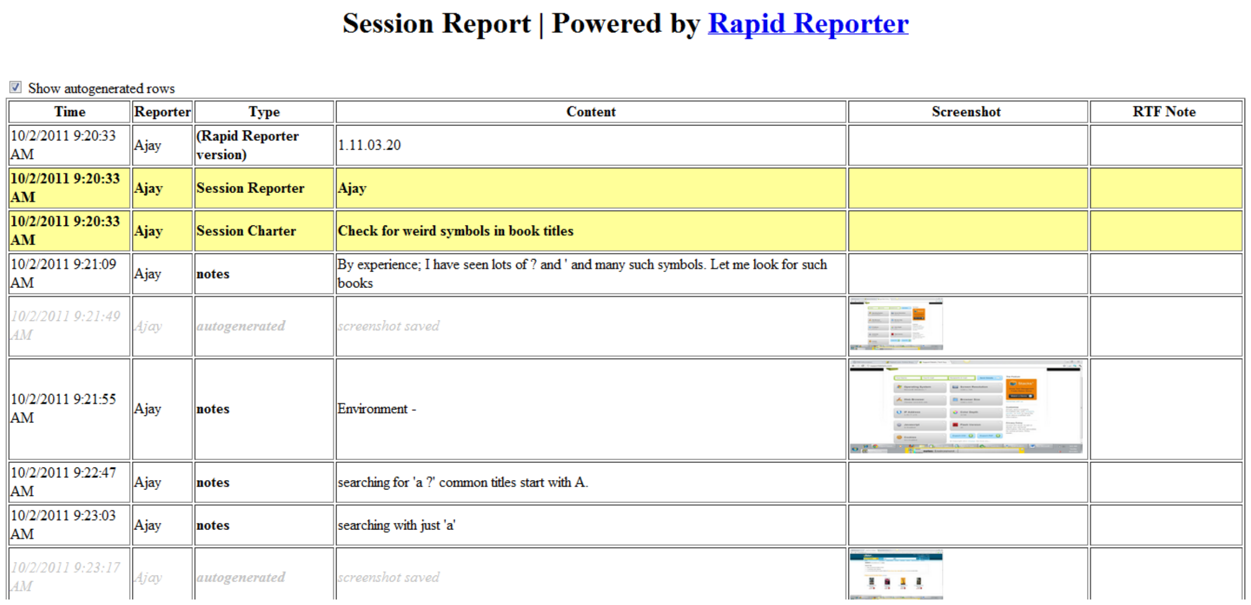


Figure 20: Sample Mindmap

If you want to make testing notes, use Rapid Reporter http://testing.gershon.info/reporter/. Rapid Reporter is a tool developed by Shmuel Gershon. It is a light weight tool with notes section and timer too. You can use it to take screenshots too. Once you are done with your testing session, review your notes and report bugs.



Session Report | Powered by Rapid Reporter

☑ Show autogenerated rows

Time	Reporter	Type	Content	Screenshot	RTF Note
10/2/2011 9:20:33 AM	Ajay	(Rapid Reporter version)	1.11.03.20		
10/2/2011 9:20:33 AM	**Ajay**	**Session Reporter**	**Ajay**		
10/2/2011 9:20:33 AM	**Ajay**	**Session Charter**	**Check for weird symbols in book titles**		
10/2/2011 9:21:09 AM	Ajay	**notes**	By experience; I have seen lots of ? and ' and many such symbols. Let me look for such books		
10/2/2011 9:21:49 AM	*Ajay*	*autogenerated*	*screenshot saved*		
10/2/2011 9:21:55 AM	Ajay	**notes**	Environment -		
10/2/2011 9:22:47 AM	Ajay	**notes**	searching for 'a ?' common titles start with A.		
10/2/2011 9:23:03 AM	Ajay	**notes**	searching with just 'a'		
10/2/2011 9:23:17 AM	*Ajay*	*autogenerated*	*screenshot saved*		

Figure 21: Sample Rapid Reporter notes

Let me sum up the points on documentation:

- Read all documents related to the product.
- Concentrate on few sections only if the document is huge.
- Pay special attention to release notes and bug fixes.
- Use a light weight note taking tool like Rapid Reporter.

Deadlines

One of the most important points to consider after bug reports is deadlines. No matter how good your bug reports or testing sessions are, they are of no value once the contest is finished. Pay special attention to the deadlines. A few tips from my experiences are as follows:

Tip 33: Start early.

Start early. It might be an obvious tip but many fail to follow it. By starting early, you have access to the application early. No bugs are logged, so any bug you log has less of a chance of being a duplicate of another bug. If you face any problems in accessing the application or bug database, it can be resolved quicker when you start early. Usually the contest owners are accessible for some time just after the contest is announced. Make use of the time by registering early and starting early.

Some contests place a limit on the number of testers. If the limit is reached, the contest automatically refuses to accept new entries. Do not let that happen to you. As soon as you join in, browse through all the document links. Check if the documents exist and if the credentials provided by the contest owner work.

Tip 34: Schedule your testing sessions.

Know the deadline and schedule the sessions. Divide your time for learning about the application, few simple tests, browsing about similar products, reading the documentation, testing different features and for logging bugs. Planning is very important in testing contests where every tester is competing against fellow testers. You may like to have a task list to help you win the testing contest. A sample task list is as follows:

Task Name	**Date**	**Status**
Check all links, user credentials	Nov 2	Completed
Read user guide, website	Nov 3	In progress
Test against checklist	Nov 3	To be done
Create mindmap	Nov 4	To be done
Test for three sessions	Nov 4	To be done
Log bugs	Nov 3, 4	In progress

Table 1: Sample task list

Having a plan helps you focus on a task whole-heartedly. Without a plan, there are chances that you might miss a task or complete it quite late. I suggest you have a simple plan and not spend a lot of time creating a detailed plan. Remember that the plan is for your reference and not for the contest owner.

Tip 35: Beware of time zone differences.

We have many contests running all over the world and the contest owner might be from a totally different time zone as yours. Make sure you understand the deadline and the time zone. Think of a deadline like Oct 03, 10:00 PM. According to which time zone? IST, CST or GMT? It might be 10:00 PM in India, early morning next day in Australia and 10:00 AM in some part of the USA.

Make sure you mark your calendar with the actual deadline. It is better if you finish your testing a few hours before. Include the deadline information according to the correct time zone in your plan. An easy way to mark a deadline is to convert the time in terms of GMT time. I refer to the GMT clock at http://www.greenwichmeantime.com.

Tip 36: Look out for constant updates.

As soon as testers start logging bugs and the contest owner wants to convey some information, the contest owner will send out an email or update the project requirements on the contest page. Many testers forget to return to check the contest page. Maybe the contest owner found out that testers are testing a blocked feature, or on a different site, or on an incorrect environment.

Sometimes the contest owner is happy with the testing results and decides to end the contest one day in advance. Modify your testing approach based on the updates. If you are not aware of the regular updates, you may miss out and lose a chance to win the testing contest.

Here is the wrap-up of tips on deadlines:

- Start early.
- Schedule your testing sessions.
- Know the time zone of the deadline.
- Do not miss regular updates from the contest owners.

Commenting on other bugs

Remember that you are the not only tester. There are many other testers participating in the same testing contest. Apart from a great opportunity to learn from other bugs, do not forget that it is a contest. You are competing. When I participate in a testing competition, I follow two simple tips.

Tip 37: Appreciate good bugs.

You might not log all the best bugs. There might be a tester who logs bugs better than you. He/she might have brilliant bug reporting skills. Appreciate such bugs. Add a comment to the bugs. Mention what you liked in the bug. Make a note of what you learnt.

Look for features where you can apply this new learning. By appreciating the testers who report good bugs, you make someone else happy. Who knows, both of you might write a book together and work on a common project after few months. It is a good icebreaker to make new friends.

Tip 38: Identify duplicate bugs.

I do not know of any contest which accepts duplicate bugs. The contest owner validates and marks the duplicate bugs as invalid after the contest ends. Sometimes the contest owner might start validating as soon as the first bug is logged. As you go through all the bugs, you might recognize a few bugs to be duplicate of other bugs. Add a comment with the original bug number.

By identifying the duplicate bug, you help the contest owner. Remember that if you mark a few bugs as duplicate incorrectly, you invite comments from the bug's tester. Your credibility as a tester will also be reduced.

It is quite simple:

- Appreciate good bugs.
- Identify duplicate bugs.

Communication

Many people are interested in the success of the product. The contest owner is not the only person who wants to see a quality product released to the market. Talk with everyone who has information about the product and who can help you win the contest.

Tip 39: Talk to other stakeholders.

With the growth in social media, it is easy to get in touch with other stakeholders. You can browse to the 'About Us' page and contact them over Twitter, email or chat. Not everyone responds to such requests. At the same time some might give you vital information about the application as they want you to help them. You never know unless you try.

I have personally experienced programmers helping me with clarifications, explaining how a particular feature is built, and other such information. All you need is a good social presence and you are all set.

Tip 40: Talk to the live support team.

Some applications support a live chat feature. Some technical personnel from the company will be ready to answer your queries and make a note of your issues. If it is possible to provide a resolution, they will be more than happy to share it with you. If there is some functionality missing from the application, they will let you know. Such conversations might give you valuable test ideas.

Once I had a chat with a live support team during a testing contest. I wanted to upload a set of 40 questions to their application. I had to email him an excel sheet with the questions in a particular format. It was a usability issue. No user was aware of the format. As I had talked to the live support team, I knew the format. How will other users know?

You can also request the team to help you with test data or any other such request as they have direct access to the code and the database. You can ask them to clear up all the data related to a particular user name or import data from a particular template.

Tip 41: Talk to other testers and users.

Just a casual chat with other testers might help both of you identify the traps each one of is falling into. After participating in multiple contests, there will be usually a group of testers who will be participating regularly. Chat with them. Share your thoughts and a few test ideas too. Beware of testers who waste your time by asking questions which can be found by a single google search. www.lmgtfy.com is the perfect site for such testers.

There are users who play a very important role in any business. Try to talk to any users who are even a little familiar with the product. Some users might post their experiences on their blog. Contact them to talk about their experiences, what kind of problems they faced, what they liked the most and which features they use the most. The time spent communicating with users will be very valuable to you as a tester.

Tip 42: Talk to the contest owner.

The contest owner is the one who has drafted the project requirements and is validating the bugs. He/she is the one who has direct access to testers' bugs. Try to contact and get to know the contest owner. Their feedback is really valuable as it is based on real time data. He/she is the closest to the actual testing competition. Some feedback from the contest owner could be about features not tested, incomplete bug reports or non-reproducible bugs.

By applying the real-time feedback, you can modify your testing strategy on the spot. Your chances of winning the contest increases based on the interactions with the contest owner.

To sum up the tips on communication:

- Contact the other stakeholders.
- Chat with the live support team.
- Interact with other testers and users.
- Talk to the contest owner.

Social Media

It is hard to count the number of social media sites a user is exposed to these days. One can start with Twitter, Facebook, LinkedIn, the different technical forums, and blogs. Information - good or bad - spreads within minutes across the internet. Photos and videos are shared, users comment on posts, recommend a product, or share their experiences on websites.

As a tester, why not use these sources of information?

Tip 43: Actual customer issues.

Many products have created their account on Twitter, Facebook, Google Plus and LinkedIn apart from having their own blog. They are constantly interacting with their customers and prospects through one of these social sites. As a tester, we can know about the complaints from the customer, the latest news about the product, feedback from the customers and questions from users. We can also understand how the product is distributed and what claims are made on the website.

Follow the product on Twitter, Facebook and blogs. By focusing on the actual customer issues, we can learn to test on similar lines.

Tip 44: Know the user scenarios.

Users don't use the product. They have a problem and see the product as a solution to their problem. They pay a lot of money to use the product. Studying the user comments helps us learn more about the user scenarios. It is a totally different experience to learn from actual customer usage than to sit in the board room designing tests. Interact with the users and know how they use the product.

Tip 45: Google for known issues.

Apart from customers, testers find issues and report them on their blogs or websites. Google for such known issues. Some of the product websites list out the known issues to help the users. Report such issues too if the features are under the scope of testing contest. There have been contests where a huge list of bugs has been found by a simple google search.

Any information about the product is valuable and helps refine your testing strategy.

To sum up,

- Learn from actual customer issues.
- Know the user scenarios.
- Use the power of Google.

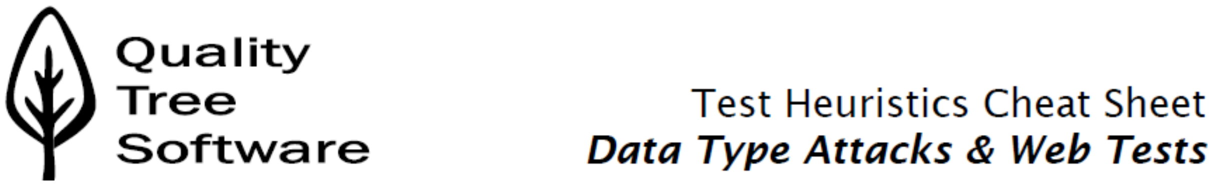
Other Information Sources

When testing under extreme time pressure, quick sources of information are handy. If you have cheat sheets, books or blogs highlighting a particular testing technique or strategy, you might have a slight advantage if you know how to use them effectively.

Tip 46: Use cheat sheets.

Use cheat sheets or checklists to assist in your testing. One of the useful cheat sheets is by Elisabeth Hendrickson and can be found at

http://testobsessed.com/wp-content/uploads/2011/04/testheuristicscheatsheetv1.pdf



Quality Tree Software

Test Heuristics Cheat Sheet

Data Type Attacks & Web Tests

Figure 22: Test Heuristics Cheat Sheet

As you complete parts of your testing, you can strike out sections in the cheat sheet or checklist. They are light weight documents and extremely helpful in contests with extreme time pressure.

Tip 47: Books, blogs and websites.

There are many books on different testing approaches and techniques. You can find a lot of books on security testing, usability testing, performance testing and mobile testing. Read them and apply the lessons. There are many blogs and forums that deal with a particular testing strategy in detail over many blog posts. Subscribe to such blogs. Quick Testing Tips blog is one such website with lots of tips on software testing.

QUICK TESTING TIPS

YOUR DAILY FEED OF SHORT SOFTWARE TESTING TIPS…

Figure 23: Quick Testing Tips

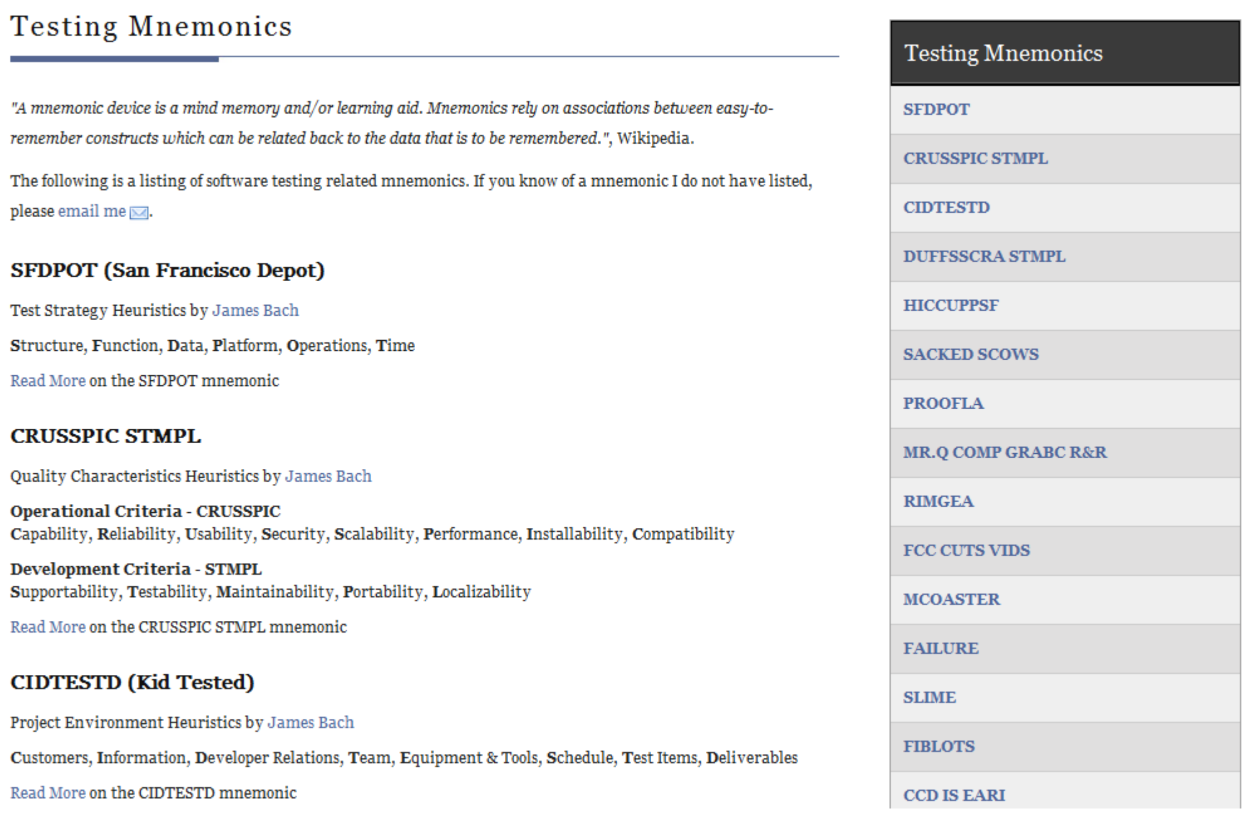
Tip 48: Similar products - comparable products.

When I test a particular product, I look for similar products on the internet. What are other products that serve the same purpose? If I had to choose two more products of the same category, which one they would be? Which product is the best in the same category? Which product is considered the standard among all the users? For example, I am testing Flipkart. I will use the test results from Amazon, eBay, Jumadi as oracles too.

Many testers fail to recognize the different sources of information on the internet. Use them testers!

Tip 49: Use Mnemonics.

As I have already mentioned in this book, many testers fail to recognize the power of mnemonics. Lynn McKee has done a wonderful job collecting all the available mnemonics on a single page. Have a look at her list at www.qualityperspectives.ca/resources_mnemonics.html.



Testing Mnemonics

"A mnemonic device is a mind memory and/or learning aid. Mnemonics rely on associations between easy-to-remember constructs which can be related back to the data that is to be remembered.", Wikipedia.

The following is a listing of software testing related mnemonics. If you know of a mnemonic I do not have listed, please email me.

SFDPOT (San Francisco Depot)

Test Strategy Heuristics by James Bach

Structure, **F**unction, **D**ata, **P**latform, **O**perations, **T**ime

Read More on the SFDPOT mnemonic

CRUSSPIC STMPL

Quality Characteristics Heuristics by James Bach

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

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

Read More on the CRUSSPIC STMPL mnemonic

CIDTESTD (Kid Tested)

Project Environment Heuristics by James Bach

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

Read More on the CIDTESTD mnemonic

Testing Mnemonics

- SFDPOT
- CRUSSPIC STMPL
- CIDTESTD
- DUFFSSCRA STMPL
- HICCUPPSF
- SACKED SCOWS
- PROOFLA
- MR.Q COMP GRABC R&R
- RIMGEA
- FCC CUTS VIDS
- MCOASTER
- FAILURE
- SLIME
- FIBLOTS
- CCD IS EARI

Figure 24: Testing Mnemonics

To recap the tips on other information sources:

- Use cheat sheets.
- Read testing books, blogs and websites.
- Use information from similar products.
- Use mnemonics.

General Traps To Avoid

Based on my experience in participating various testing contests, here are some common traps:

Tip 50: Testing on the live site instead of demo site.

Last month I tested a website for creating surveys. I made a mistake. In between my tests, I browsed to the actual site. The next time I launched my browser, I was testing on the actual site instead of the demo site. I went a step ahead and logged a bug found on the live site. Only when the contest owner marked the bug invalid, did I realize my mistake.

Beware of this kind of mistake.

Tip 51: Testing out-of-scope features.

When we test any application, some features attract us with a lot of bugs. You might discover a lot of bugs but are you sure that the feature is not an out-of-scope feature? All your efforts would be of no use if the contest owner marks the bugs as invalid with the comments - 'Out of scope - Do not test.'

Make a list of features which need not be tested. Anytime you test the feature by mistake or you have doubts, you can refer back to the list.

Tip 52: Assuming and not questioning.

Many testers still lack questioning skills. They assume even if there is an option to question and get information. Unless you question, others will not know your thought process. As testers, you are hired to help them take decisions by providing information. If you have any questions, feel free to highlight them to the stakeholders. In testing contests, you can email the testing owners or the one who provides you the credentials. Once you ask them and clarify your questions, it helps in identifying the right tests for this contest.

Some traps:

- Testing on the wrong version or environment.
- Testing out-of-scope features.
- Assuming and not questioning.

After The Contest

If you have followed the above tips, I am pretty confident that you have given your best and there is a good chance of winning a contest. What next after winning the contest?

Here are a few additional tips:

Tip 53: Understand the test ideas.

Now that there is no time pressure, go through each bug slowly. Understand the test idea used. Note how the tester has composed the bug report. Think of ways in which you can improve the bug report. Think why you missed the bug. Remember not to miss similar bugs and test ideas.

It is a good idea to make a note in your tester's notebook. You can refer to it at regular intervals till you remember all the test ideas without referring to any book.

Tip 54: Contact other testers.

Once the contest is over, congratulate the top testers. Contact the testers with whom you would like to stay in touch. Make new friends. Interact with them, talk about test ideas, other testing contests and their hobbies and interests. Exchange contact information. If you reside in the same city, try to meet. It is a totally different feeling to meet personally and exchange testing ideas.

Tip 55: Thank the organizers.

Thank the organizers - the contest owner, the stakeholders who responded to your queries. Building relationships is one key task of any professional. Thank them for their hard work and support. Who knows, you might be invited to test a private project. Don't thank expecting a favor. Thank for the opportunity and support you received from them.

Tip 56: Provide overall feedback.

Provide your honest opinion about the product. Include both the positives and areas of improvement. Will you use it? Which features are the most difficult to use? Which user scenarios will you use the most? Include them in the feedback. It actually helps the contest owner and other stakeholders.

Tip 57: Make notes from the contest.

What did you learn from this contest? Did you win? If yes, why do you think you won? What did you do best? Which aspects need improvement? If you did not win, why? What can you do better next time?

Note everything in your testing book. Re-read, do not repeat the mistakes and keep winning!

To sum up tips on after the contest:

- Understand the test ideas.
- Contact other testers.

- Thank the organizers.
- Provide overall feedback.
- Make notes from the contest.

Best of luck for the contest. Keep winning and learning from each contest. ☺

I hope this book is useful to you and your team.

If you have any comments on any of the chapters or the book in general, feel free to contact me. I will be more than happy to discuss any of the topics.

My contact details:

Email: ajay184f@gmail.com

Skype/Twitter/GTalk: ajay184f

LinkedIn Profile: http://in.linkedin.com/in/ajaybalamurugadas

```
☺☺☺☺☺ ☺   ☺  ☺☺  ☺   ☺ ☺  ☺ ☺☺☺☺
  ☺   ☺   ☺  ☺☺  ☺☺  ☺ ☺ ☺  ☺
  ☺   ☺☺☺☺☺ ☺☺☺☺ ☺ ☺ ☺ ☺☺   ☺☺☺☺
  ☺   ☺   ☺ ☺  ☺ ☺  ☺☺ ☺ ☺     ☺
  ☺   ☺   ☺ ☺  ☺ ☺   ☺ ☺  ☺ ☺☺☺☺
```

Ajay Balamurugadas