

# Managing Your Focus While Exploratory Testing

- *Experience Report by Tina Zaza* -

*11/21/2009*

## 1. *Preparing Yourself/What to Test*

Before getting yourself in "the mood" to start exploratory testing, first you'll want to gather your resources.

- Consult the tests you've already done that were considered mandatory and (hopefully) covered by a requirement (this will help you to know where you left off and what not to include in your exploratory tests). This can be helpful if you have multiple testers on your team with the work divided amongst you. You may not have been aware of all the tests someone else has already done.
- Consult a developer or a business/systems analyst to better understand what you're going to be testing and the variations (or "What if's") that might be available to you. If a developer is worried about the requirements being too vague in a certain area, focus on those requirements and look for gaps/misunderstandings. If they are concerned that a scenario might break their code on the slim chance it ever occurred, try it out to see exactly what happens, and check if that risk is acceptable to the business.
- Create a list of "What if's" and other scenarios that you, yourself, plan to cover. This is also a fall-back if the developers and analysts don't have much to suggest.
- Take a break! You've just done your planning!

## 2. *Getting Started*

Once you have your plan together and your preliminary questions answered, go to it!

- Block off time on your calendar for the next hour or two. You could even book yourself a conference room so as to not be interrupted. Also, you might consider closing your email.
- Open all applications, matrices, databases, etc that you'll need to do your testing. As dumb as it sounds, stopping to open additional documents and tools right when you need them actually slows down the process.
- Do as much on your list as you can in the time allotted, noting any questions you have along the way.
- If you want to add tests that you might think up as you go, create yourself a new list for those. If they are easily done on the current track you're on, go ahead and finish those up. If they are completely separate from what you're working on, leave them for the next "session"
- One approach is to test a lot in a short time frame: Try to work on the simplest/quickest tasks first ("quick wins"), and check them off your list. Then go to the more complex tasks that will take longer to do. This will also make you feel more accomplished at the end of your day!
- Another approach is to test more important scenarios: Do those first! You can tell which scenarios might be more important during your planning steps. Ask the developers and analysts if they were "worried or concerned" about any certain conditions. If time allows, and I realize it hardly ever does, have the developers look over your original test cases (this is something that may take place the first time you test the requirements, are a part of your company's process). They may see something missing or request that you try something in addition to your tests, if time allows. That's what you want to hear, because those "if time allows" tests are a great starting point for exploratory testing.
- A third approach lacks any consideration for time and importance, but can be just as effective: Create areas of interest, say one application function, or a bunch of similarly

related tasks. Big or small, work on all those pieces together. They should all tie into one another, which will save time that might be spent switching between applications and tools to test possibly unrelated “quick win” tasks.

- After your allotted time is up, try to stop. Consider it an accomplishment. Check your list for whatever you didn't get done, and add it to the next session's tasks. Also add to the list anything you came up with in addition to these tasks. Schedule yourself time for the next day or time block and repeat the process for steps 1 and 2.
- Take another break! Since exploratory testing is often more in-depth than your everyday testing, try to only work on it in small spurts. This will help your brain come to a stopping point and refresh itself before going back in for another one or two hour session. You won't get overwhelmed all at once, and you'll be able to focus attention in one place instead of trying to multi-task. The easier and more straight-forward each task, the better.

### 3. *Avoiding Interruptions*

Everyone knows that getting good, heads down work done takes time, patience, and focus. You don't want to be stopped from your work once you start. (Or maybe you do, but I don't know how you get anything done that way!) Note: These suggestions are for everyday, not just while exploratory testing.

- Try to separate yourself from your distractions. Tell dependent co-workers that you'll be busy for a while and that you'll get back to them later. Turn off your instant messenger, cell phone, and email (if that's feasible). Leave your desk to go work somewhere else. Schedule yourself a meeting.
- While you're testing, if you find something that does not look correct or seems odd, write it down, take a screenshot or document it however you seem fit, and move along. You might come back to it at the end of your testing to see if it's a repeatable issue.
- While you're testing, if you find something that needs immediate attention, you could call over a developer or analyst (since they wouldn't be distracting if they're helping you!) to clarify. You may need to stop what you're doing to create a defect. Remember that this is still productive, even if you have to stop to write up a ticket! If it's something that is not urgent and/or obviously doesn't require much work to fix (an example is on-screen wording), then mark it on the list to create a defect or otherwise address the issue when you're finished.
- If someone does find you in the middle of your testing, kindly tell them you can't talk at the moment unless it's an emergency, and that you'll get back to them as soon as you're done. Also suggest that they send you an email with their question/comment/issues/whatever, so that they don't forget, and so that you can have the email to view at your convenience.
- For phone interruptions, they created voicemail and Caller ID for a reason.
- For email interruptions, if it was urgent, they would have walked over or contacted you through IM/phone.
- For some people, it's helpful to wear headphones at work with or without music. It keeps a lot of other sounds out. For other people, it's a distraction in and of itself.
- Remember that anything you can get done is better than nothing!