Chapter 1. You don't see any elephants around here, do you?: What Do-It-Yourself Usability Testing is, Why It Always Works, and Why So Little of It Gets Done

> *Why are you waving that chicken around over your head?* To keep the elephants away. *Does it work?* You don't see any elephants around here, do you?
>
> —VERY OLD JOKE

OK, before we get to "do-it-yourself usability testing," first, what is "*usability testing*"?

It's pretty simple:

> Watching people try to use what you're creating/designing/building (or something you've already created/designed/built), with the intention of (a) making it easier for people to use or (b) proving that it is easy to use.

There are a lot of different "flavors" of usability testing, but the one thing they all have in common is that they involve watching people actually *use* the thing.

This element of actual use is what makes usability testing very different from things like surveys, interviews, and focus groups, where you're asking people for their opinions about things, or their past experiences using things.

One useful way to categorize all the different flavors is by thinking of them as either quantitative or qualitative.

In a **quantitative** test, you're interested in *proving something* ("Is this latest version better than the previous one?" "Is our site as easy to use as our competitors' sites?") and you do this by *measuring things* like success rate (how many people finish the tasks you give them to do) and time-on-task (how long it takes them).

Since the purpose is to prove something, quantitative tests are like scientific experiments: They have to be rigorous, or the results won't be trustworthy. This means you have to define a test protocol and follow it consistently for all of the participants.[1] You have to collect data carefully. You have to have a large enough sample of participants to make your conclusions statistically significant, and they need to be representative of your actual users so you can extrapolate the results to a larger population. All of this means you have to know what you're doing, and you have to be careful doing it.

[1] *In usability testing, we call the people we're observing "test participants," not "test subjects," to remind ourselves that we really aren't testing them—we're testing the thing they're using.*

In quantitative testing, you usually try to minimize the amount of interaction with the participant to avoid influencing the results. In an extreme form (sometimes called "Voice of God" testing), the participant sits in a room by himself with a facilitator giving him instructions over an intercom, while an observer watches through a one-way mirror and records the data.

So, what's *"Do-It-Yourself Usability Testing"*?

As you might have guessed by now, the kind of testing I'm recommending you do is at the opposite end of the qualitative–quantitative spectrum.

"Do-it-yourself" usability tests are definitely **qualitative.** The purpose isn't to *prove* anything; it's to get insights that enable you to *im*prove what you're building.

As a result, do-it-yourself tests can be much more informal and, well, unscientific. This means you can test fewer users (as long as you get the insights you need), and you can even change the protocol mid-test. For instance, if the first participant can't complete a particular task and the reason why is obvious, you can alter the task—or even skip it—for the remaining participants. You can't do that in a quantitative test because it would invalidate the results.

Basically, a facilitator sits in a room with the participant, gives him some tasks to do, and asks him to think out loud while he does them.

There's no data gathering involved. Instead, members of the development team, stakeholders, and any other interested parties observe the session from another room, using screen sharing software. After the tests are finished, the observers have a debriefing session where they compare notes and decide what problems should be fixed and how to fix them.

That's really about all there is to it.

The funny thing is, it just works

When I teach my usability testing workshops, I always begin by doing a live demo test—"live" in the sense that it's completely unrehearsed. The only preparation I do is to choose a site that belongs to one of the attendees and use it just long enough to come up with a task that I think people are likely to want to do on that site. (For example, if it's a health care site, I might make up a task about booking an appointment.)

Then I ask for a volunteer to be the test participant and spend 15 minutes doing an abbreviated version of a test. (Real tests typically last about an hour, although they can be as short as five minutes and as long as an entire day.)

The result is almost always exactly the same:

- The participant has a good time and gets a round of applause at the end for being brave enough to volunteer.
- The site's "owner" spends the entire 15 minutes furiously scribbling notes about things to fix and asks if she can have the recording to show to her team and her boss.[2]

[2] *One "owner" wrote me a few months later to tell me that after viewing the demo test of his site, his team had immediately made one simple change that they calculated—based on the data from the first few months—was going to save their company $100,000 a year. (It had to do with getting customers to sign up for online billing.)*

- Everyone else ends up thinking, "Gee. Is that all there is to it? *I* could do that."
- When it's over, I ask, "Does that seem like a worthwhile way to spend 15 minutes?" and everyone nods their head in agreement.

The point of doing the live demo is to show people that (a) there's nothing to it, and (b) it always works. I can tell that some of them suspect that I'm able to make it look easy because I've done it a lot. But by the end of the day, after everyone has tried conducting a test themselves, they all seem to understand that there's no magic involved and that it really is as easy as it looks.

I have to admit I was a little anxious the first few times I did live demo tests for an audience. But I've probably done fifty of them by now and it's worked every time, no matter what the site is and no matter who the participant is.

The fact is, it just works. Ask anyone who's done any amount of usability testing and they'll tell you that it pretty much always works. If you sit somebody—almost anybody—down and have them try to use what you're building, they'll inevitably encounter some of the problems that most people are going to encounter.

But *why* does it work?

It may not seem to make sense that something so simple (just giving people something to do and watching them do it) can consistently reveal serious usability problems. But if you think about it for a while (or for several years, in my case), there are reasons why it works:

- It works because all sites have problems. We all know this from our own experience. How often have you used a Web site and not run into a usability problem? And they're often significant problems that seriously frustrate you or even keep you from doing what you set out to do.
  Some mature sites may have fewer serious problems, especially if they've been through repeated rounds of usability testing, but don't kid yourself: Your site has usability problems. Heck, *my* site has usability problems, which as you can imagine is potentially quite embarrassing. Even Amazon has usability problems, and it's common knowledge how highly I think of Amazon.[3]

  [3] *People love to email me about problems they find on Amazon.com, as though I could do something about them. I do have an Amazon Prime membership ($79 a year gives me "free" second-day shipping), but that's about the extent of my influence. And Amazon does so much usability testing that if there's a problem, I'm sure it's not because they're not aware of it; they probably just haven't decided what to do about it yet.*

- It works because most of the serious problems tend to be easy to find. Again, think about the usability problems you've run into on other people's Web sites. Don't you usually find yourself thinking "How can they possibly *not* know about this problem?" Many of the most serious problems are lying around in plain sight, and almost everybody will run into them.

And yet on our own sites, we somehow think of them as being hard to find. It always reminds me of the Vietnam-era Doonesbury cartoon where Phred asks the curator of a demolished Cambodian museum if it was destroyed during the secret bombings.

DOONESBURY © 1973 G. B. Trudeau. Reprinted with permission of UNIVERSAL PRESS SYNDICATE. All rights reserved.

I KNOW, BOY, I KNOW! I WAS THE CURATOR.
YOU WRETCHED SOUL! DID THIS HAPPEN DURING THE SECRET BOMBINGS?
SECRET BOMBINGS? BOY, THERE WASN'T ANY SECRET ABOUT THEM! EVERYONE HERE KNEW! I DID, AND MY WIFE, SHE KNEW, TOO! SHE WAS WITH ME, AND I REMARKED ON THEM!

The usability problems on your site may not be obvious to you, because you know how it works—or how it's supposed to work. Most of your users, on the other hand, don't, and that makes all the difference.

Of course, there are also serious usability problems that are more “hidden,” the kind that not as many people will run into. But unless you have substantial resources to devote to usability (for instance, it’s your full-time job), I strongly recommend focusing on getting rid of the obvious ones first. Most sites don’t even manage to accomplish that.

And finally:

- It works because watching users makes you a better designer. Even though terms like “user-centered design” and “user experience” are now in the vocabulary of most people working on Web sites, relatively few designers, developers, stakeholders, managers, and check-signers—who all have a hand in the design process—have actually spent any time watching how people use Web sites. As a result, we end up designing for our abstract idea of users, based for the most part on ourselves.
  Watching users makes you smarter about how people use things and how things can be designed for use. I like to say that it informs your design intelligence, sort of the way travel is a broadening experience.

Why so little of it gets done

So, if it's so easy and so valuable, why isn't frequent usability testing a standard part of every Web project? Even today, very few organizations do any usability testing, and if they do, they usually only do it once, near the end of the project.

I think it's largely because most people still haven't had any firsthand experience with usability testing, so they don't know how valuable it can be. But even if they have, there's no shortage of plausible reasons not to do it.

Lack of time, for instance. Testing seems like a lot of work, and most of us already have more on our plate than we can manage. Most Web development schedules are so tight that the prevailing attitude is "Let's get it out the door, and we can tweak it later."

And then there's the natural—and nearly universal—reluctance to show our work before it's finished. We always know that what we're working on has problems, so why bother showing it to people and wasting our time (and theirs) having them tell us what we already know? (And who likes having the flaws in their work exposed in public, anyway?)

These are all quite reasonable, but as you'll see, they're not necessarily true.

FAQ

You're talking about very small samples. Can't we get more reliable information from things that gather data about a lot of people, like Web analytics?

Yes, Web analytics can give you a very accurate picture of what people are doing on your site ("72% of all visitors left the Home page after less than 5 seconds"). The sample size is very large (all of your users, in fact), the data is based on actual use, and the query tools allow you to pose almost any statistical question and get an answer immediately. And with the advent of Google Analytics at such an attractive price point (free), this kind of data is available to everyone.

The problem, though, as any usability professional will be happy to tell you, is that while analytics can tell you in great detail what people are doing on your site, they can't tell you *why* they're doing those things. For instance, if people are spending a lot of time on a particular page, the statistics can't tell you whether it's because they found the content very useful and they're busy reading it or because it makes no sense and they're busy trying to figure it out.

Usability testing, on the other hand, excels at helping you understand why people are doing things.

When it comes to finding and fixing usability problems, if I had to choose between awesome analytics that could tell me exactly what my users are doing (but with no chance to know what they're thinking while they're doing it) or sitting with one user for an hour, with the ability to hear what he's thinking and ask probing questions, I'd take the one user every time.