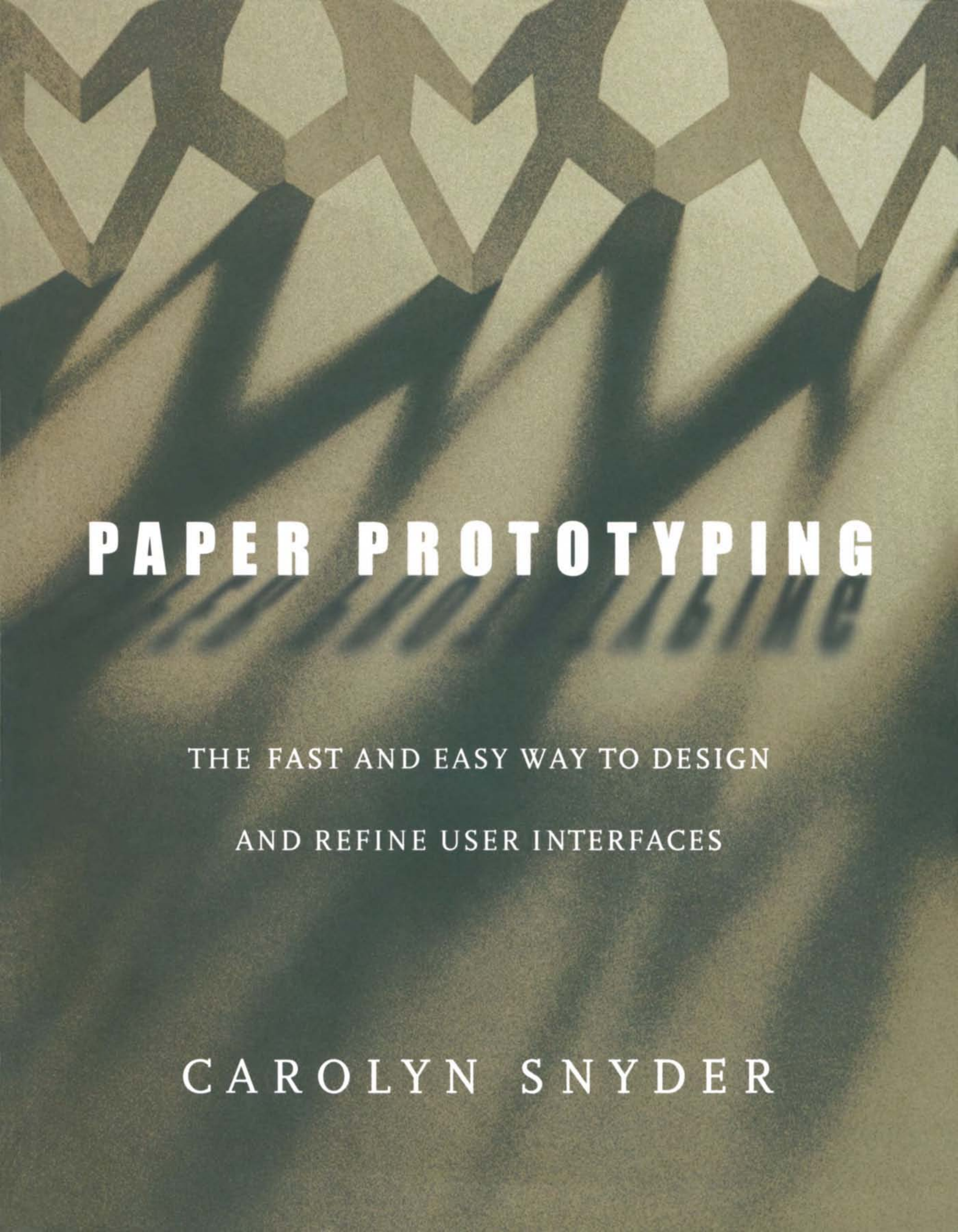


PAPER PROTOTYPING

THE FAST AND EASY WAY TO DESIGN
AND REFINE USER INTERFACES

CAROLYN SNYDER

Chapter 8

Introduction to Usability Test Facilitation

It wouldn't be very useful to sit a user down at a table covered with prototype pieces and ask, "So what do you think?" People have a hard time answering abstract questions like that, especially out of context It's important to not just ask users what they think, but to see what they do. It's like the difference between looking at a car on the showroom floor and taking it for a test drive.

To conduct a usability test, you need to have users, realistic tasks, and a test facilitator to run the show in a structured yet informal manner. This chapter explains what the facilitator does during a usability test. If you're already experienced in conducting usability tests, you may want to skip to the next chapter.

In a nutshell, the purpose of the usability test facilitator is to maximize the amount of useful and reliable data from the test sessions while at the same time minimizing the stress on the users. As you might expect, these two goals can conflict with each other, so a good facilitator understands how to make the trade-offs.

Are You Dangerous?

I believe that anyone who has good social skills and a genuine interest in usability can learn the basics of facilitation well enough to be effective, although as with any skill, there are nuances that may take years to master. There's a risk that this chapter may give you just enough information and encouragement to be dangerous—fools rush in where angels fear to tread. One chapter can't do justice to all the special situations that can arise or explore the interesting debates among usability professionals about how to handle them. Before facilitating your first test, I recommend that you watch a few tests run by an experienced facilitator, do additional reading (see the References section), or take a training class. Join an organization like the Usability Professionals Association *(www.upassoc.org)* or SIGCHI *(www.acm.org/sigchi),* which may have chapters in your area. Practice your skills whenever possible and ask colleagues to give you feedback.

Facilitator Responsibilities

Here's an overview of the activities typically performed by the usability test facilitator:

- Before the users arrive, brief the in-room observers on how to behave, what to watch for, and how to take notes (Chapters 10 and 11 discuss this further).
- Greet the users, brief them, obtain their informed consent, and pay them.
- Escort users to the test room and introduce them to the Computer and any observers.
- Ask users to introduce themselves and summarize their background.
- Explain the testing protocol—how to interact with the paper prototype, work together, think aloud, and so on.
- Facilitate each task, interacting with users as needed.
- Manage the time spent on each task, covering the areas that are of greatest interest to the observers.
- (Optional) Facilitate a short Q&A session with the users (and observers, if present) to discuss interesting issues that arose during the session or ask questions not covered by the tasks.
- End the session on time.
- Thank the users and escort them out.
- Debrief the observers—list issues and make changes to the prototype in time for the next test. (This can be done by a core team member instead of the facilitator.)

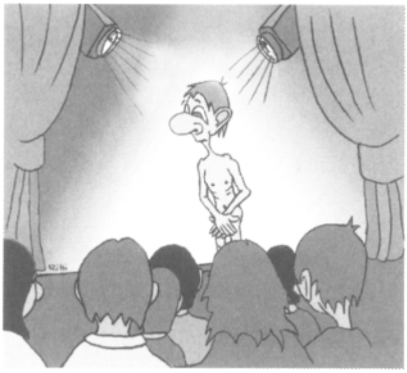
Ethical and Legal Responsibilities

The goal of usability testing is to learn how to make the product better with the assistance of users—but not at their expense. As the facilitator, you have an ethical and legal responsibility to make sure that your test participants do not have an unpleasant experience.

Many years ago, the spell checker in Microsoft Word (I think it was version 2.0) did not recognize the word "usability" and suggested "suability" instead. I was highly amused at the time—I was writing a report that compared the usability of a Microsoft application to one of its competitors—but it's stuck in my head ever since as some kind of subtle cosmic warning that usability testing can land you in legal hot water.

Central to the idea of usability testing is the concept of informed consent—the user understands the nature of the session and their role *before* agreeing to participate. With the possible exception of paper cuts, there is little risk of physical harm during a paper prototype usability test. However, any kind of usability testing carries the risk of psychological or emotional harm in the form of embarrassment, frustration, or stress or feeling stupid. It's your responsibility to avoid causing this kind of harm. Just as conscientious backpackers leave their campsite cleaner than they found it, the facilitator's goal is to have users leave the session in as good or better frame of mind as when they came in. Of course, users may be more tired, but they will also have reasons to feel good, such as hearing that their perspective is valuable and how their feedback will make the product better.

As of 2002, there is no formal code of ethics for usability testing, but it is likely that one will be developed in the future. In the meantime, you may want to study the ethical guidelines used in related professions like psychology. See *www.paperprototyping.com* for more information.



This isn't how you want a usability test to feel!
(Illustration by Rene Rittiner.)

The Informed Consent Form

The informed consent form is a contract between you and the user that states you've explained the purpose and nature to their satisfaction, including the following:

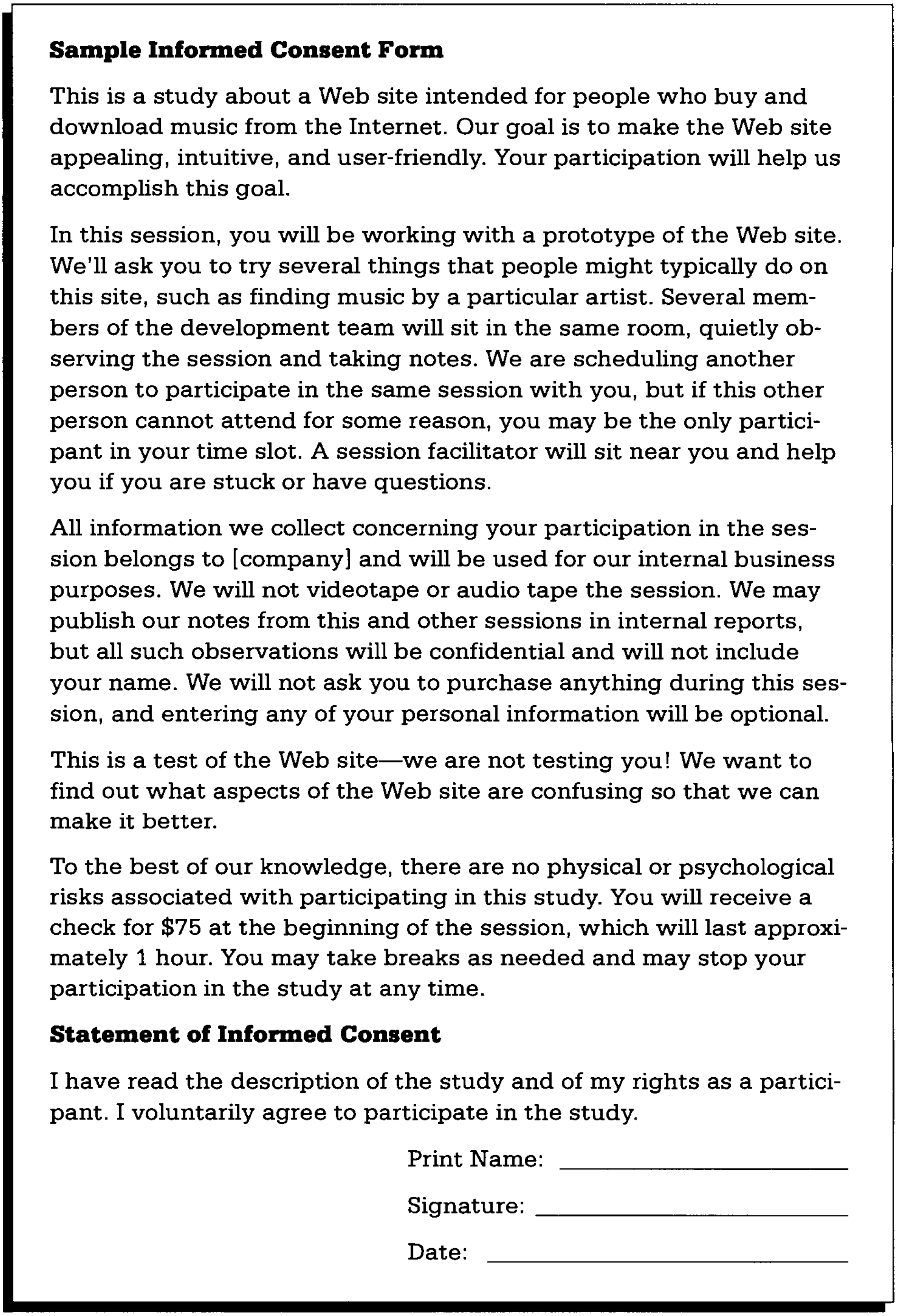
- What you'll ask them to do (i.e., work with a prototype)
- What's being tested (the interface, not the user)
- Any risks to their physical or psychological well-being (such as the presence of observers)
- The length of the session, including their right to end it at any time
- What data you will collect and how this information will be used

Ideally, the informed consent form should be sent to users ahead of time so that they have a chance to read it, contact you with any questions or concerns, and decline to participate if they are not comfortable with the test setting. If observers will be present, the informed consent form should explicitly mention that fact. If the session is being videotaped, the form should explain what the tape may and may not be used for. However, just because users receive the consent form doesn't mean they will read it, so the facilitator should go over this information when the users arrive for the session. (The next chapter contains some sample wording.)

Although the informed consent is a legal document, it shouldn't be written using legal jargon. Write in plain English so that users can form an accurate picture of what the session will be like and what's being asked of them. Figure 8.1 shows an example of an informed consent form. A document containing the most recent version (I keep evolving it as I learn more) is available on *www.paperprototyping.com.*

Nondisclosure Agreements

Depending on your circumstances, you may need to ask users to sign a nondisclosure agreement (NDA). An NDA is a legal document that prohibits participants from discussing what they saw in the usability test unless or until your company makes that information publicly available, for example, by announcing the product. NDAs are typically used when the product is new or undergoing substantial changes and the company doesn't want to leak the details. But in practice, many companies don't use an NDA for usability testing because there is little risk of anything bad happening if users discuss what they saw (make sure your recruitment



Sample Informed Consent Form

This is a study about a Web site intended for people who buy and download music from the Internet. Our goal is to make the Web site appealing, intuitive, and user-friendly. Your participation will help us accomplish this goal.

In this session, you will be working with a prototype of the Web site. We'll ask you to try several things that people might typically do on this site, such as finding music by a particular artist. Several members of the development team will sit in the same room, quietly observing the session and taking notes. We are scheduling another person to participate in the same session with you, but if this other person cannot attend for some reason, you may be the only participant in your time slot. A session facilitator will sit near you and help you if you are stuck or have questions.

All information we collect concerning your participation in the session belongs to [company] and will be used for our internal business purposes. We will not videotape or audio tape the session. We may publish our notes from this and other sessions in internal reports, but all such observations will be confidential and will not include your name. We will not ask you to purchase anything during this session, and entering any of your personal information will be optional.

This is a test of the Web site—we are not testing you! We want to find out what aspects of the Web site are confusing so that we can make it better.

To the best of our knowledge, there are no physical or psychological risks associated with participating in this study. You will receive a check for $75 at the beginning of the session, which will last approximately 1 hour. You may take breaks as needed and may stop your participation in the study at any time.

Statement of Informed Consent

I have read the description of the study and of my rights as a participant. I voluntarily agree to participate in the study.

Print Name: ____________________

Signature: ______________________

Date: __________________________

Figure 8.1 Sample Informed Consent Form.

process screens out any competitors), and sometimes the company even considers it a benefit if test participants start spreading the word. Discuss the need for an NDA with your Legal or Marketing department, or other powers that be.

If your company has a standard NDA document, it'll likely be more complicated than what you need for usability testing—most NDAs are written for contractors or companies who are potential business partners. Participants in usability tests don't receive proprietary materials such as functional specs or business plans, so some of the stipulations may not apply. On occasion I've seen a user back out of a usability test due to discomfort over an imposing legal document. If users must sign an NDA, ask if it's possible to use a simplified version that merely prohibits users from discussing what they see until the product is released.

Payment for Users

Users who have no association with your company will expect some sort of payment for their time. Expect the compensation to be in proportion to the specialization of the expertise you're looking for. In one Boston-area study in 2002, I paid network administrators $125 for a 1-hour usability test of a specialized Web application, and in another study I gave consumers $75 for a 90-minute test of a travel Web site. Talk to a market research firm to find out the going rate in your area and whether nonmonetary gifts are appropriate—a bottle of good wine might be appreciated in one culture but taboo in another. Those little marketing giveaways (a key chain or coffee mug with your corporate logo) are nice extras, but they're not a substitute for more substantive compensation.

Users are entitled to their full payment simply by virtue of showing up on time, even if you discover that a person does not meet the screener (that's the fault of the recruiter, not the participant). I like to pay users at the start of the session—it starts things off on a pleasant note, and I believe that it may reduce the perceived pressure to perform—but usability specialists are still debating the merits of paying before versus after, and as of this writing neither approach is considered wrong.

Decide on the form of payment before recruitment begins. Many participants expect cash by default. If you are paying with a check or gift certificate, make this clear up front in case the user is expecting to use the cash for cab fare home. Consider reimbursing users for transportation or parking if those expenses are nontrivial. Unless you plan to pay users more than a few hundred dollars, you probably don't need their tax ID number, but double-check this with your Accounting department. When I use cash payments, I have each user sign their name, date, and amount received on a sheet of paper, which serves as a record of what happened to all that cash.

So far I've been discussing test participants who have no association with your company. But payment may not be appropriate if users are employees or customers of your company or if they work for a government or industry that regulates the maximum gift that can be accepted—even a coffee cup may be considered too valuable (I wish I was joking about that, but I'm not). Talk with your Human Resources or Sales department to determine what is appropriate for the user profile you've selected.

Facilitator Roles: Flight Attendant, Sportscaster, Scientist

A smoothly run usability test can appear deceptively simple.* A test facilitator is like a duck—serene on the surface, but paddling like heck underneath. There are many judgment calls that the facilitator makes in every usability test. How long should you let the users struggle when they're stuck? When and how should you give them hints? What do you do when there's not enough time left to complete the task? The guidelines I provide here will help answer those questions, but it will take practice for you to build confidence in your skills.

A test facilitator is like a duck—serene on the surface, but paddling like heck underneath.

In understanding what it takes to be a good facilitator, I've found it helpful to think in terms of three roles:

1. Flight attendant—safeguard the physical, psychological, and emotional well-being of test participants.
2. Sportscaster—maximize the flow of information from the users to the observers.
3. Scientist—maintain the highest possible degree of integrity in the data.

These roles capture the essence of the facilitator's responsibilities. As you'll see later, they also provide guidance for the inevitable situations when trade-offs arise.

* A friend once asked me point-blank after helping me with a pilot test, "Companies *pay* you to do that?"

The Flight Attendant

(Jack Hollingsworth/Getty Images.)

The flight attendant is the most important of the three roles. Just as a real flight attendant's primary responsibility is the physical safety of passengers (in the early days of aviation, flight attendants were trained nurses), the facilitator makes sure that the testing experience is emotionally nonthreatening for participants. In the flight attendant role, the facilitator is responsible for pretest briefing and obtaining informed consent, monitoring the users throughout the session for signs of stress, and providing reassurance and assistance as needed. Handing out those little packets of peanuts is optional.

Comfort is important too. All usability tests should take place in an atmosphere of hospitality, which helps relax the user. A flight attendant role is a service-oriented one—the test facilitator should greet users, hang up their coats, offer a beverage, and perhaps chat for a few minutes. These small courtesies establish the facilitator in the users' minds as a helpful person who cares about their well-being. I do not recommend having an assistant perform the greeter tasks and then bring the users into the test setting where the facilitator awaits; although this may be appropriate in a scientific setting, paper prototyping is an informal and "high-touch" activity, and it works best when the facilitator and users have established a rapport. If circumstances require the use of a greeter, coach him or her in the particulars of the flight attendant role.

The flight attendant also notes each user's demeanor—are they cheerful or serious? Assertive or timid? What kind of day have they had so far? Although it's impossible to make a thorough assessment of someone's personality in a few minutes, the more you can learn about a user, the better you will be able to make judgment calls when that user is stuck. I might joke with a user who was confident and upbeat, but with a person who seemed shy or tired I would avoid humor and be a little quicker to step in when he or she ran into difficulty.

The flight attendant has duties after "take off" as well. Throughout the test, the flight attendant monitors users for signs of stress. Some red flags include sighing, short answers to questions when longer answers were previously given, apologies or other self-blaming statements, and so on. The flight attendant remembers to offer a mid-test break (often appreciated by the observers as well) and a beverage refill. If a user ever appears distressed, the facilitator has the right and responsibility to pause or end the test session. (This situation should be very rare—one in a few hundred tests—so a full discussion is beyond the scope of this book. Suffice it to say that the facilitator's primary responsibility is to salvage the ego of the participant.)

Once I have thought of the situation in terms of the ***information*** *that is lacking, it's easier to say something to the user that sounds respectful and reassuring but not phony.*

One of the most important things that the flight attendant can do is assure the users that they are holding up their end of the bargain, even when—make that especially when—they encounter difficulty. Although we'd like it if everyone sailed through the tasks, it's more valuable for the product team to find the things that give people trouble. When users get stuck, they're really doing us a favor, albeit one that might feel painful to both them and us. It's important to acknowledge and alleviate that pain in a way that comes across as respectful. This is sometimes tricky—gratuitously positive feedback like "Gosh, you're doing great!" or "Don't worry, it's not your fault" may sound hollow or patronizing to a struggling user. There's also a risk that praise can make users think they're on the right track when they're not.

I find that it helps a lot if I refuse to think of users as dumb or ignorant. It's much more accurate, not to mention fair, to think in terms of specific knowledge that we mistakenly assumed users would have. If a user-blaming thought enters my mind (for example, "He has no clue how to do this"), I recast the issue in terms of the missing facts: "He's shown us that people need to know X, Y, and Z to do this task. We thought this user profile would know X from their jobs—maybe our assumption was wrong. And we really do need to find a better way of explaining Y and Z." Once I have thought of the situation in terms of the *information* that is lacking, it's easier to say something to the user that sounds respectful and reassuring but not phony.

It may take quite a bit of practice before it feels natural to provide feedback in an appropriate way. Over the years, I have learned to take my cues from the users.

Although some people might want reassurance that you don't blame them for the problem, others merely need to hear that you've followed the logic of what they're doing. Here are several different examples of the kinds of things I've said to users who encountered difficulty. What all of them have in common is their respectful intent based on my assessment of the user and the situation. You'll need to decide which of these examples might be appropriate for your users, and over time you'll come up with your own.

"This is exactly the kind of feedback we were hoping to get."

"It's not just you. Other people have run into this too, so it's definitely something we need to change."

"What you've done makes perfect sense, so hopefully we can change the logic of the system to support your approach."

"Thank you—you've just found something important that we wouldn't have noticed on our own."

"The developers are intimately familiar with the code, and sometimes they don't realize that they've omitted crucial information from the interface."

"You're part of our market for this product, so your perspective is valuable."

"Hmm, it never explained how to do X, did it? In real life it wouldn't be fair to expect people to use this without being told about X."

"Thanks for hanging in there . . . it appears that we made this way harder than it needed to be, and I'm sorry about that."

"This is very helpful—you're doing just what we need you to do."

Although the flight attendant is the top-priority role, it is usually not the role that occupies most of the facilitator's time. As long as all is going well, the flight attendant stays in the background so the sportscaster can have center stage.

The Sportscaster

While the flight attendant's attention is focused on the user, the sportscaster serves the observers, who are members of the development team. The main responsibility of the sportscaster is to ensure that the observers get as much useful information from the test as possible.

(Ron Case/Getty Images.)

Thinking Aloud versus Talking with Users

A common technique for conducting usability tests is the *think-aloud protocol,* where you ask the users to articulate their thoughts as they work on the task. But let's face it, thinking aloud isn't what most of us do. If someone went through their daily life explaining their every action, we'd question their sanity. So although it's fine to ask users to think aloud in a usability test, most of them aren't going to do it perfectly and it certainly can't be called natural. You know what is much more natural? Talking with users as they work. Especially in a paper prototype test, this turns out to be the way to go.

Here's the rationale. In paper prototype testing, the Computer is sitting right across from the users, so their first tendency will be to talk to the Computer. But you don't want this because then the Computer will feel a social obligation to help the user by explaining the interface. So as the facilitator, you should be the person the users talk to, and the easiest way to do that is by talking to *them.*

But do so carefully. Rather than agreeing, disagreeing, or explaining, the facilitator should be asking questions, encouraging users to elaborate, and remaining neutral toward the interface and its designers. It's easy to inadvertently give users a clue about what they're supposed to do. Confirming afterward that the users made the correct choice isn't quite as bad, but should still be avoided. So before speaking, consider what effect it might have on the users' behavior.

You may have heard the story about Clever Hans, a horse that supposedly could count by reading a number off a card and tapping his hoof that many times. Except it turned out that Hans' proud owner (and also the researchers who tested this supposedly psychic horse) was subconsciously giving him the cue to stop tapping. If horses are smart enough to play this game, it's a safe bet that humans are too.

The Play-by-Play

Although in theory the observers see and hear everything that the facilitator does, there are some reasons why this may not be true in practice. The facilitator usually has the best vantage point because he or she sits close to the users and the prototype. People who are sitting farther away or in another room may not be able to read the error message that just came up or see which menu option the users clicked on. Observers who are writing notes may have missed what the users just did. Some users speak softly, and observers can't hear them.

So one thing the sportscaster does is what I call the play-by-play—verbally reinforcing any user action that might not have been obvious or visible to observers. For example, if the user writes something in the search field, you'd say, "So you typed 'return policy' in there." And then when the Computer hastily scrawls a "No matches found" error on a scrap of paper, you'd say, "Hmm, 'no matches found.' What does that mean to you?" Obviously, you don't want to mindlessly parrot every detail—the sportscaster uses this tactic judiciously, when there's a good chance that observers missed some information that's important to understanding the users' behavior or context. It can alter the user's behavior if you call attention to something the user hadn't noticed, so it's best to do the play-by-play for those things you're pretty sure the user is already focusing on.

Here are some other sportscaster tricks I've learned over the years.

Encourage Questions, But Don't Answer Them

One of the ironies of the test facilitation is that you want to encourage the users to ask questions, but most of the time you don't want to answer them, at least not right away. This can feel awkward, so you might want to prepare some responses ahead of time. For example, if a user asks you the meaning of a term, you might say something like, "That's a really great question but forgive me if I don't answer it yet—maybe you'll discover the answer as you go." Other tactics are to direct users' attention back to the task ("Why is that important?") or to plead ignorance ("Hmm, I'm not sure.") But do acknowledge the users' questions in some manner, or they may stop asking.

Write down all the questions that users ask. Questions often indicate something important about the functionality or usability of the interface. At the end of the test, if the users didn't find the answer to something they were truly curious about, you can enlighten them. ("Remember that undo function you wanted? It was hidden over here.")

Use the Users' Vocabulary

The words you use when talking to users can influence their behavior. Avoid the temptation to use the "right" terms—when possible use the same vocabulary that the users have already used. Otherwise, you might inadvertently provide a clue about how the interface is supposed to be used. For example, if a user refers to the corporate logo on your Web site as "the beach ball," you would call it the beach ball, too, not the home page link.

Use Open-Ended Questions

It's usually better to ask open-ended questions rather than closed-ended ones because the former encourage users to give more detailed answers. The purpose of an open-ended question is to encourage users to reveal their underlying thought process.

> "What will that do?"
>
> "What are you trying to do right now?"
>
> "What are you thinking?" (Use a neutral tone with this one!)
>
> "Hmm, tell me more about that."
>
> "What does ____ mean to you?"

Although open-ended questions are often preferable, sometimes you may want to use a closed-ended question if time is short or you really do want to know something specific, such as, "Did you see this link to the return policy over here?" However, I suggest reserving such questions for the end of the session because they can change the user's subsequent behavior. In addition, you should be careful not to imply that the user has missed something "obvious."

Listen for Nonspecific Utterances

Vocalizations such as *hmm, ah, oh,* or *oops* usually represent the tip of a cognitive iceberg, and they're your cue that something important is going on inside the user's skull. Ditto for nonverbal gestures. A person who is confused or thinking may wrinkle a brow, frown, or put a pen in his or her mouth. All these cues offer a great opportunity for the facilitator to probe what's happening in a nondirective manner: "John, what's going through your mind right now?"*

Make Use of "Hourglass Time"

Paper prototype tests often have short pauses when the Computer is looking for (or creating on the fly) the next piece of the prototype. Instead of sitting in silence, you can use this time by summarizing the actions the users have taken so far (or better yet, ask them to do so) or by asking what they expect will happen next. It also might be a good opportunity to offer a break.

Learn When to Shut Up

Unlike a real sportscaster, it's not necessary or even desirable for the facilitator to keep up a nonstop barrage of questions and commentary. As a new facilitator, I sometimes pummeled users with so many questions that they didn't have sufficient time to consider their answers. When people are thinking, it's perfectly appropriate to let the silence go on for a bit—15 or 20 seconds will seem a lot longer to you than to them—before you encourage them to verbalize. When a person is conversing, his or her brain cannot be fully focused on the interface. If you're always jumping in with questions or hints, the development team may complain (with some justification) that you aren't giving the users enough time to figure things out on their own. In other words, you're part of the problem.

Let Users Decide When They're Done

As a rule, you should let the users decide when the task is done. For example, I once had a user give up on a data entry task—the information was saved automatically, but he thought he had to use the Save command, which happened to be grayed out. After 10 minutes of frustration, he gave up. This was an important usability issue, but one we wouldn't have seen if we had stopped him the moment *we* knew the task was complete.

End Tasks Early if Appropriate

As an exception to the preceding rule, on occasion you may want to end a task prematurely if the part you care about happens early. For example, on an e-commerce site, you might want to watch users complete the checkout process all the way to the confirmation screen on one task, but you don't need to see this more than once. On the next task, when the users add the item in the shopping cart, I might say, "I'm going to ask you to pause here. You've covered the part we needed to see,

* Just don't get too carried away. I once commented that a user was scratching his head, where-

so in the interest of time we'd like to move on to something else." (This is just my preference, but the word "pause" feels less awkward to me than "stop," which implies that I didn't like what the users were doing.) Caveat: Use this tactic sparingly (ideally no more than once per test) because users may become inhibited if they expect to be stopped at any moment.

Consider Allowing Between-Task Discussion

My usual practice is to allow observers to ask a question or two at the end of each task, although some facilitators may choose not to do this if they're concerned about maintaining control of the session. But I keep these discussions brief, usually just a couple of minutes, because one of the facilitator's responsibilities it to manage how the test time is spent. If an issue has come up that merits more in-depth discussion, I might give the observers the choice, for example, "We have about 20 minutes left. Do you guys want to continue this discussion now or do one more task and then come back to it?" Although I'm still managing the time, the observers get to decide which activity is of greater interest to them.

The Scientist

The scientist is responsible for the integrity of the data, through note-taking, videotaping, written tasks, test procedures, and so on. The scientist strives to maintain objectivity and to minimize any adverse effects on the data being collected. People who are unfamiliar with usability test facilitation may mistakenly think that this is the facilitator's primary purpose, but in paper prototype usability tests, the scientist role is usually the third priority—there are many circumstances where the scientist takes a back seat to the other roles.

I don't want to imply that being scientific isn't important; professionals in any field should always understand their methods, apply them appropriately, and seek ways to improve them. But keep in mind that paper prototyping is not a technique that's used when precise usability measurements are required. It's more of a blunt instrument. Most of the time, there isn't a need to calculate statistics. You don't use an eye-tracker or spend hours analyzing videotapes. Most of the data from a paper prototype usability test is qualitative, not quantitative.

Is objectivity important? Yes and no. Clearly, the facilitator has the opportunity to introduce all kinds of bias into the testing process, through something as obvious and deliberate as a hint or as subtle and unintentional as a smile when the user makes a correct choice. It's virtually impossible to remove the human element from paper prototype testing. Thus, the scientist's focus isn't on minimizing interaction with the users but rather on monitoring how it might affect the results.

(Getty Images.)

For example, if the users are given a hint, the scientist makes note of it so that the team can discuss later how the users' behavior might have been affected. For example, "They found the answer in the help, but only after a suggestion that they look there. This is evidence that the help *content* works, but we can't conclude whether users in real life will see it." However, it is usually best for the facilitator to avoid asking leading questions or revealing his or her own opinions of the interface, so the scientist strives to maintain this objectivity. Chapter 13 revisits the topic of bias in more detail.

For beginning facilitators, it is often hard enough to fulfill the flight attendant and sportscaster responsibilities without adding note taking to the mix. I recommend that new facilitators skip the note-taking until they're comfortable managing the action. Instead, invite plenty of observers and have them take notes (which they should be doing anyway). I also know some facilitators who feel, even after several years of experience, that they do their best work when they leave the note-taking to others.

Co-Discovery (Two-User) Testing

Usability testing with two users at once is not, strictly speaking, a paper prototyping technique, but most of the tests I've conducted have been co-discovery because it seems to work quite naturally. In a co-discovery session, the two users work on the tasks together, discussing them with each other (and the facili-

Benefits

There are several reasons why co-discovery can be useful:

- **More comfortable for the users.** A solo user who gets stuck might blame himself or herself, but a pair of users who are stuck realize, "Okay, obviously this isn't just me." Especially when you have in-room observers, having users work in pairs can help them relax.
- **Easier for the facilitator.** Contrary to what you might think, facilitating a test with two users is often easier than working with one. In co-discovery, users talk to each other as they work through the tasks, and the facilitator may end up saying relatively little. This is fine. With only one user, the facilitator usually needs to take a more active role and be more conscious of providing positive feedback.
- **More data.** With two users, you're getting two people's thoughts and reactions at the same time. Note that you're not necessarily getting two *independent* viewpoints—User B may quickly agree with what User A said, making it hard to determine what User B really thinks. But if both users provide a plausible rationale for why they agree, that can give you more confidence about the need to change something. And when users disagree, then you usually do have two valid data points. Although this isn't very scientific, I tend to split the difference and view co-discovery as providing about 1.5 times the data of a single-user test.
- **Scheduling.** In several hundred usability tests, I estimate that I've experienced an average no-show rate of about 1 in 10 users. If you've scheduled two users and only one shows up, you can still hold the test. That's useful if you're testing on a tight schedule or have observers traveling to attend the sessions. Double no-shows are very rare—perhaps 1 in 100 tests—so with co-discovery there is little chance of having to cancel a test.

Drawbacks of Co-Discovery

Although it has its advantages, there are also some drawbacks to co-discovery:

- **Discrepancy in experience or confidence.** If one user knows substantially more than the other (or thinks so anyway), it can be awkward for the other person. Ideally, you'd avoid this during the recruitment process by not pairing people with a mismatch in experience, but this makes recruitment more com-

- **Peer pressure.** Some participants may be more concerned about looking foolish in front of someone else in their profession than they are in a roomful of strangers that they'll never see again.
- **Slower pace.** Sometimes two users will take longer to complete tasks because there's more discussion. This drawback balances the "more data points" benefit—it's a trade-off.
- **Dominant personality.** Most people are pretty good about sharing and will work out for themselves who does the clicking and typing. Some pairs will automatically switch roles after a task or two. But a few people don't get good marks in "plays well with others" and tend to dominate the session. This doesn't happen very often, but when it does, it makes the facilitator's job harder.

Why I Choose Co-Discovery

I've found that co-discovery works well for paper prototyping—I use co-discovery about half of the time in nonpaper testing, and more like 90% of the time when testing a paper prototype. (I'm usually testing on a tight schedule, plus it's easier for two people to share a paper prototype than a computer.) In usability studies where I've done both single-user testing and co-discovery, most of the time the product team has felt that they got more benefit out of the co-discovery tests. But this is a generalization, and there are some usability specialists who believe that the drawbacks of co-discovery outweigh its benefits. Although the rest of this book assumes that you're testing with two users at a time, I suggest that you try both methods and decide which works best for you.

Friends or Strangers?

There are some advantages in pairing up people who don't know each other. Usability guru Jared Spool tells the story of how he once facilitated a test with a married couple who spent the entire session communicating in half-sentences:

She: "Why don't we click on the . . ."

He: "I was just thinking that . . ."

She: "Oh yeah, you're right—let's . . ."

Jared dubbed this communication shorthand "married people's syndrome," but it can happen with any two people who know each other well. (There's also an ominous variation where the users know and *dislike* each other.) Strangers tend to be more thorough about discussing things, so their conversation is easier for observers to follow. It's also easier to avoid inequities in power—a manager/subordinate combination is especially risky.

Although I usually try to pair up strangers, this is not a hard and fast rule. There are some situations in which it makes sense to recruit people who know each other:

- The product is typically used by people who know each other (for example, a parent and child, or people who are playing a game together).
- The users are coming from some distance away (perhaps one of your customers) and want to share a ride.
- Users indicate that they would feel more comfortable working with someone they know.

To make the friends-or-strangers decision for your product, consider the circumstances of its use and what will be most comfortable for the user population you've chosen.

Making Trade-Offs

Let's go back to those roles of flight attendant, sportscaster, and scientist. When various situations arise in usability tests, it's often helpful to frame the situation in terms of a trade-off between two of the roles. Usually the flight attendant role has the top priority, but there are times when you might decide to let the sportscaster or scientist have their way. Thinking in terms of roles may help you understand the opposing forces at work so that you can make the best decision for each situation. Following are several examples.

Situation: Users are stuck on a problem you haven't seen before.

This common situation is a flight attendant/sportscaster conflict. The sportscaster wants the development team to get as much information as possible about the issue, including everything that's going on in the users' heads, but the flight attendant wants to make sure that the users don't feel stupid or embarrassed. Usually it's okay to wait and see if they can get past the difficulty on their own, but the flight attendant should provide reassurance and be prepared to intervene if necessary. (If your users are familiar with the game show *Wheel of Fortune*, you might ask, "Would you like to buy a vowel?" A bit of gentle humor can diffuse tension, and it's also a subtle reminder that the user is in charge.) Later in this chapter, I provide more tips on getting users unstuck.

Situation: *Users are stuck on a problem that also came up in the last two tests.*

Naturally, the flight attendant is concerned and should remain vigilant, but this is primarily a sportscaster/scientist trade-off. The main question to ask is, "How much data do we need about this problem?" If the team is already painfully aware of the issue, there's little point in spending valuable test time covering the same ground again. In this case, the scientist might give the users a hint to get them over the difficulty, noting the fact that he or she has done so. But if not all of the observers were present at the previous tests or there's no consensus about the severity of the problem, the sportscaster might overrule the scientist to let the observers get more information about the problem.

Situation: *Two users working together disagree on what to do next.*

The sportscaster might want the users to go down the wrong path first because it will likely be more interesting to watch them realize the error and recover from it—one hallmark of a usable interface is that it helps users get back on track. But the flight attendant must approve because it would be detrimental to foster conflict among the users and facilitator. One way to direct the users is by saying something like, "You guys have different ideas about how to proceed, which is fine because people do things differently. Why don't you try what John suggested, and if that doesn't do what you wanted, you can switch to Mary's approach." The scientist notes that the users were given this instruction (I call it a "nudge") about which way to go first. However, if time is short and there's additional ground to cover, the scientist may overrule the sportscaster by suggesting that the users take the correct path.

Situation: *The team has redesigned a screen, but the users don't take the path that will lead them to it.*

This is another situation in which the sportscaster will probably take priority over the scientist, although not immediately. The sportscaster knows that the team is eager to find out how their revised screen works, but the scientist first wants to establish whether the users would have gotten there on their own. So the scientist will avoid taking the users to the redesigned screen at first, until it's reasonably clear that they won't go there. Then the scientist can step aside in favor of the sportscaster, who directs the users to the new screen to find out how they react to it.

Other Common Testing Challenges

Writing about everything that can possibly go wrong during a usability test would be a book in itself. But here are some common situations and tips for handling them.

Getting Users Unstuck

It's usually not a question of whether users will get stuck, but rather how soon and how badly. Having users run into difficulty while attempting tasks is perhaps the most valuable part of usability testing because it indicates a problem with the interface that you can (hopefully) solve now that you're aware of it.

Once you explain something, you forever lose an opportunity to understand the problem.

When users ask a question, resist the temptation to answer it. Once you explain something to a user, you *forever lose an opportunity to understand the problem.* Once the answer has been revealed, users may have one or both of the following reactions:

- They will be embarrassed at what they perceive to be their own ignorance and not want to tell you what they were thinking. "Oh, never mind, it was just me."
- They literally can't reconstruct or articulate what their thought process was in the absence of information that they now have. They'll say something like, "Okay, that makes perfect sense. It's fine the way it is."

On the other hand, you don't want the test to grind to a halt. My favorite tactic for getting a user unstuck is to ask a series of questions, starting very general and progressively getting more specific. For example, say you're testing a telephone interface and the user can't figure out how to transfer a call:

Facilitator	*User*
1. What are you trying to do right now?	I want to transfer this call to Mike in Accounting.
2. What do you think the next step is?	I want to dial 5385, but not lose this guy I'm talking to.
3. (A small hint): Do you see anything that might help you?	I'm not sure . . . if I dial the extension, won't it just beep?
4. (A big hint): What do you think the Flash button does?	I was wondering about that, but I was afraid it would hang up.

This approach is useful in ferreting out the root cause of a problem. You verify that the user:

- Is trying to do what you thought (perhaps the user assumes the caller would prefer to be given Mike's extension instead of being transferred).
- Shares your understanding of how to go about it.
- Can find and use the specific function or control needed to proceed with the task.

In this example, the root cause of the problem is insufficient information about how the Flash button works.

The User with an Agenda

Every once in a while a user will take advantage of the usability test as an opportunity to give the product team detailed—perhaps repeated—feedback on what's wrong with the interface or how it "should" be designed. Sometimes when users have behaved this way, it's been my fault for not briefing them well enough on their role. A user who has participated in focus groups may believe that they're being asked to give their impressions of what they like and dislike, which is usually not the primary goal of a usability test. An excess of opinion can also happen if the user is an influential stakeholder, such as an important customer.

Users who have interface design backgrounds are often tempted to offer their suggestions because they believe—sometimes even correctly—that their expertise is valuable to the team. Unless you're looking for a design consultant, it may be helpful to screen out potential test participants who claim to know a programming language, have created a Web page, and so on, unless of course these users are your target audience. (This is similar to the reason why people who work for market research firms aren't allowed to participate in focus groups—they know too much about the methodology, and it affects their behavior.)

Regardless of the cause, when you encounter a user with an agenda, there are a few tips for keeping him or her focused on what will provide the most value to you:

- Avoid asking the user's opinion about likes or dislikes—you'll just open the floodgates.
- Direct the user's attention back to the interface, for example, "Please show me what you'd do next."
- Let the user know you've captured his or her feedback—sometimes the person just wants to be sure the message has been heard. One way to confirm this is, "Thank you, I've made a note of that." If the user makes the same point again shortly thereafter, you might try, "Yes, you mentioned that."
- If necessary, politely curtail further discussion. My favorite way to do this is by saying, "In the interest of time, there are some other areas we want to cover." This reminds the user that there are plenty of other issues that we would like his or her feedback on.

Unusually Nervous User

Perhaps the most difficult situation a facilitator will ever face is a user who simply can't manage to relax. I have seen initial nervousness in such a variety of usability testing situations that I find it impossible to generalize about the cause—male/female, solo/co-discovery, paper/computer, lab/living room, with/without videotaping, a dozen/no observers. I believe that a few users are destined to experience initial nervousness in virtually any test setting, and we should accept this as unavoidable even as we seek ways to alleviate it.

But every now and then, a user who starts out uncomfortable stays that way. If a user doesn't relax after he or she starts interacting with the paper prototype, here are some tips:

- Provide help sooner—don't let the person struggle.
- Take a break and assess how the user is feeling. End the test if necessary.
- Give positive feedback and reassurance as explained in the flight attendant section.

- Avoid humor, especially sarcasm. In a delicate situation, trying to be funny can give users the mistaken idea that you're laughing at them. I have learned the hard way that flippant comments I aimed at the interface ("Oh yeah, like that makes sense") can strike users instead.

Mismatched Users

Another challenge in facilitating co-discovery usability tests is when one user has less confidence than the other does. (Sometimes this user has less knowledge of the subject matter, but not always.) Try to get feedback from both users. Don't allow an outgoing or confident user to speak for both—ask the other user whether he or she agrees or disagrees, or even direct questions to that user first. But if one user seems especially reticent, avoid putting him or her on the spot.

Tips for New Facilitators

Facilitating usability tests can feel awkward at first. Here are some of the things I found most helpful when I was first starting out.

- **Use a checklist.** A checklist is a bulleted list of everything you want to cover in your briefing of users, introduction of paper prototyping, and so on. A good checklist is generic enough to be used in a variety of usability studies, so you shouldn't have to keep rewriting it. (The next chapter has some examples of checklists you can use while conducting paper prototype usability tests.)
- **Wean yourself from scripts.** In a script, you write down everything you will say to the users and read it to them at the proper time. Scripts are useful in usability testing when it's important to control all the interactions you have with the users. For example, if you were testing two competing software packages, you'd want your introductions of them to use equally neutral wording. But in paper prototype testing, it's unlikely that you need this degree of scientific rigor—when an interface is rapidly changing, a detailed script can become more trouble to maintain than it's worth. I've heard of facilitators who spent so much time scripting paper prototype tests that they lost all the benefit of it being a fast technique. Although scripts are okay when you're starting, as you gain confidence in usability test facilitation, I recommend that you pare them down into checklists.

- **Practice out loud.** In any kind of public speaking (and test facilitation counts!), it's helpful to practice out loud. I recommend practicing your facilitation skills with co-workers and friends—not necessarily the whole usability test, but at least the introduction and instructions. Saying everything out loud will help you feel more comfortable, plus if something comes out wrong or you find yourself at a loss for words, that's one less mistake that you'll make with a real user.
- **Seek feedback.** Although I no longer videotape usability tests (the next chapter explains why), as a learning experience there's no substitute for seeing yourself on video. (In my first experience of reviewing test tapes that I'd facilitated, I realized that I had a bad habit of trailing off and not finishing . . .) Or invite a colleague to sit in on your tests and give you feedback; one inherent benefit of the usability profession is that we strive to be empathetic rather than critical. Whenever I conduct a usability study, I ask the team afterward what they thought went well and what they'd like to do differently in the future. If I realize I've made a mistake, such as asking a leading question, I'll discuss it as an example of what not to do.
- **Strive for progress, not perfection.** One of my experienced colleagues told me that she expects to make at least one mistake per day of testing, and she relaxes once she realizes she's made it (as in "Okay, that was my stupid thing to say today. I should be fine now.") It's always a good idea to improve your skills in accordance with whatever best practices may exist in the profession. At the same time, I don't believe that any two people will ever facilitate a usability test in exactly the same way any more than two software engineers will ever write the same code. As you become experienced in facilitating, you'll learn tricks that work well for you. And despite how carefully you've prepared, try to accept that you'll also make mistakes that you can learn from.