

# GETTING READY FOR BETA TESTING



**For many companies, beta test preparation is fraught with uncertainty.**

Even knowing when you'll start can be complicated. You might want to base the decision on QA milestones, ensuring the product is stable enough for users. But if you're facing a product release window that can't be missed, your beta schedule might be set by counting backwards on the calendar. The goal of this whitepaper is to establish a set of best practices for beta readiness (private beta tests, specifically). That way, you can be confident that you're ready to launch an effective beta test regardless of any looming uncertainty.

In our experience, the best way to get ready for beta testing is to establish concrete guidelines in three areas: product readiness, team readiness, and tester readiness. If these areas are out of sync or underprepared, your beta will most likely suffer. But when they come together, they enable higher participation, easier management, and better results.

## ONE PRODUCT READINESS

The general standard for beta product readiness is a stable, feature-rich (although not necessarily feature-complete) product. If you give beta testers a very buggy product, you'll generally see a flurry of initial activity—comprised primarily of redundant bug reports—followed by a major drop off in participation. Beta testers expect that the test product will have some issues, but there's always a tipping point after which they're too frustrated to keep trying.

What we do to ensure product viability is perform a simple diagnostic check on the beta product before distributing it to testers. The check starts with determining an acceptable level of basic functionality according to the features of the product and requirements of the users. Then we install and/or power on the product, using the beta build that will undergo testing, and verify that it possesses the basic functionality we're expecting.

There are three additional things to keep in mind for these checks:

1. You probably don't want to perform your diagnostic check based on scripts from QA. They're not likely to tell you much, since the product should have been subject to those tests already. Be a little more exploratory. And perhaps this goes without saying, but you shouldn't ask QA to run these checks for you. Generally, it's best to limit their involvement to providing test equipment.
2. If you're beta testing hardware products, be sure to perform the diagnostic on a small, random sample (e.g., 20% of the units). You test multiple units so you don't later discover that the one unit you tested was the only one that worked. And you select them at random, rather than sequentially, so if any issues were introduced or fixed at some point in manufacture, you're more likely to encounter them.
3. If you're beta testing software products, you should install and perform the diagnostic check on each major platform you plan to test in beta (e.g., Windows 7, Windows Vista, Windows XP, OS X Lion, OS X Snow Leopard, Android, iOS, etc.).

At this point, your product might be viable for beta testing, but that doesn't mean it's beta-ready yet. You still need to think ahead to the end of beta testing. As they say, you can't put the genie back in the bottle. So what sort of things do you need to anticipate? Anything restricting the beta product's use, particularly after testing is complete.

Consider these three common examples:

1. Protecting beta software from file sharing with an activation/license management system
2. Automatically disabling beta software and hardware after testing is complete
3. Asking beta testers to return beta hardware after the test

In the first two cases, there's a technical component that needs to be built into the product prior to distributing it to beta testers. If you distribute without it, there's nothing you can do to protect your product retroactively. In the third example, preparing for return shipment before the start of testing is essential. You'll want to give your testers all the return shipping materials up front, as well as clear instructions about return procedures. If you don't do this at the beginning, testers are much less likely to comply. Lastly, you always want to make sure that the license you're granting testers covers the product's status at the end of testing—whatever it may be.

## TWO TEAM READINESS

When it comes to beta team readiness, you'll want to be very systematic and detail-oriented. It can be challenging at times, because you'll often be managing your own immediate team and interfacing with stakeholders in other departments (and sometimes other companies). It's worth the effort, though. Just think of how easily your beta test would veer off course if one of those stakeholders wasn't up to speed on responsibilities or schedules.

To ensure beta team readiness, try to incorporate these best practices:

- Start by defining your core parameters and processes (e.g. testing goals, strategies and mechanisms for collecting feedback, categories for bugs, incentives, etc.). Basically, make a great beta plan to guide you. It may sound obvious, but unless your immediate team solidifies these details in advance, you'll be prone to oversights throughout this process.

To help identify and record the critical details of beta testing, we've created a free set of extensive beta test planning kits for both [software](#) and [hardware](#) products.

- Create a list of all beta stakeholders, paying careful attention to individuals in other departments and external partners, with clearly defined responsibilities and schedules. Then, review your plans with each person and get their commitment.

- For any deliverables (e.g., support documentation, surveys, product keys, NDAs, the product itself, etc.), try to take receipt from the stakeholder in advance of the beta test. Otherwise, make sure deadlines are understood and any necessary resources are made available. People who aren't familiar with beta testing might not realize how detrimental even 1-2 days of waiting for a resource can be if testers are kept idle.
- For stakeholders who are responsible for directly participating in the beta test activities, make sure their calendars account for potential slips in schedule. If someone's availability is limited because of a planned vacation, the potential birth of a child, or some other event, you'll want to make sure you have a contingency plan in place.
- Lastly, for any infrastructure that will be relied upon during the beta (beta test management tools, customer support, bug tracking, content delivery, servers, etc.), make sure they are accessible, active, and tested well in advance. If any of the systems require logins for the testers, they should already have accounts and be able to log in.

## THREE TESTER READINESS

Once you've handled product and team readiness, beta tester readiness should be fairly straight forward. The key here is to make sure that your testers aren't just willing, but are also ready and able to test. From there, your responsibility is to make sure nothing is inhibiting their participation throughout the entire test.

Here are some quick tips:

- Make certain that your testers understand how to use the systems you're providing for the test. If they aren't dead simple, provide some sort of training or documentation. Friction or confusion about how to participate can squash their interest right from the outset.
- Any resources or information needed by testers to carry out their responsibilities should be both easily accessible (physically or electronically) and easy to understand.
- Have your beta participation agreements and non-disclosure agreements signed and stored before you start. Also, make sure you explain your beta secrecy requirements to testers in plain English.

If you need help in this area, we also have a free [Beta Test Agreement Kit](#) available for download. This kit includes both participation and non-disclosure agreements.

- Be clear with your testers about their responsibilities and schedule, then stick with it the best you can. By and large, beta testers aim to please and respond well to solid direction.

- Verify contact information and addresses early, especially if you're shipping any sort of physical product or incentive to your testers. Even if you're testing electronically distributed software, verifying contact information is still important. A simple phone call to an inactive tester sometimes is all it takes to spark great participation.
- Finally, when it comes to scheduling, provide an appropriate amount of time to complete tasks and other responsibilities. It's important to keep in mind that beta testers are volunteers with other responsibilities, and this isn't a full-time job.

While the ideal beta readiness guidelines for your product may vary from what we've discussed here, hopefully this guide will get you thinking and planning. That's really the best defense against the uncertainty that precedes a beta test. Accept that certain things may be beyond your control, but meticulously plan and prepare for everything else.

### Are you planning on starting a beta test soon?

If so, we'd love to give you a demo of our industry-leading beta test management platform, **Centercode Connect**. Connect provides a unified platform capable of handling all of your beta testing needs. It will help you launch tests faster, recruit better testers, keep participation high, and collect the exact feedback you need—while delivering real-time, actionable data to all the teams that need it.

**Sign up for a Live Demo or Free Trial today.**

Thanks for reading our whitepaper. We hope it was helpful. If you'd like to download our other free beta testing resources, be sure to visit <http://premium.centercode.com>.

Cover image courtesy of Flickr user **tableatny**