---

Title: Don’t listen to your users have them show you

Date: 2015-5-2020

Author: Tyler

---

How many times have you lived this story? A user has a problem with your product, and is nice enough to inform you of their issue. The problem gets vetted, and interpreted to a bug or story. This bug or story contains the problem, and a proposed fix. The diligent design and development teams follows the instructions to a “T” and fixes the bug, or creates a new feature. The user gets the feature, but doesn’t actually solve their problem. WHAT HAPPENED?!

Chances are its a classic game of telephone! I found a legitimate UI bug in twitter, so like any person interested in making the internet a better place, I hunted down their feedback/bug submitting area. I typed out a well written bug, submitted my gif, and thought that bug was off to bug triage… Shortly after I got an email saying that they didn’t understand the my bug, because their input system doesn’t take .gifs. This should’ve be an immediate red flag. If your bug intake process doesn’t take the format your users are giving you, you aren’t going to get their help.

Why should I care about visual bug reporting?

* Accuracy - Designers, developers, and product owners do our best at interpreting the never ending stream of feature requests and bug fixes that are coming from every direction. But as humans we make mistakes, we record things falsely, we hear what they want to hear. In order to cut down on these mistakes, have your users and stake holders show you the problems instead of just writing down a couple rushed words.
* Personal - When users feel as though they are contributing to a system they are more likely to develop appreciation (or cognitive dissidence) towards your app or service. Correct listening skills can transform a users bad experience into something much more meaningful and productive. Your users want to be apart of something bigger than themselves. In this day and age of never ending individualism and self promotion Maslow’s third level of needs is more in effect then ever before.
* Robust data - Users try their best to explain what they think the solution is to their problem, but as designers and developers know that isn’t always what will benefit them the most. Similar to the famous quote by Henry Ford “If I had asked my customers what they wanted they would have said a faster horse.” You know your system and what its capable of. Your users may be asking for a bigger or smaller button, when what they actually want is no button at all.

When it comes to reporting UI bugs there are several methods that are far superior to the traditional describe your problems with a paragraph of text and maybe a screenshot. Here are my favorites arranged in order of awesomeness.

* In person — If the problem was experienced by someone you can physically have a conversation with, DO IT!!! Go talk to them, sit with them, talk about the bug or feature request! This is a great opportunity to turn any simple bug fix or feature request into an impromptu focus group where you can get a feel on the pulse of your users.
* Virtual face to face — There are several online tools that offer great alternative experience to meeting face to face. Google hangouts, skype, join.me, goto meeting, etc, all have the ability to share your screen which is really the most important feature when having your users show you their troubles. Virtual meetups also have the advantage of being much faster in that you don’t have to physically move yourself from location to location. Potential downsides to this type of meeting is the classic problem of microphones and cameras not working at the beginning of the meeting.
* Gifgrabber — gifgrabber is a simple tool for OSX that allows users to capture gifs of sections of their screen, it can then easily upload those gifs to the web, so they can send you a link. I highly recommend this for situations where you need both asynchronous communication AND the ability to log problems.
* usersnap.com Usersnap is a tool that allows for easy annotation and automated capture. It will automatically pull in browser information and allow the user to comment and call out things that need some TLC.

Whatever method you chose to use, make sure that the barrier to entry is low enough that people are willing to follow through with following a bug, and valuable enough that you can solve their issue with the information they give you up front.