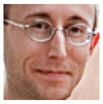
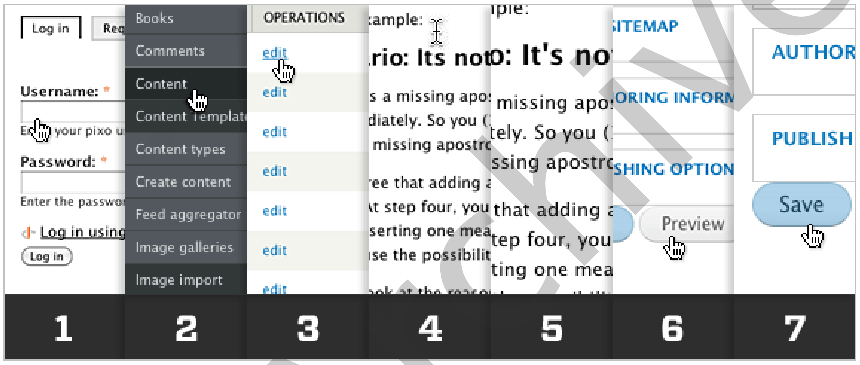
Editing the Web



By **dyoung** on September 21, 2011

Pixo is exploring a new way to edit the web with a new project that we began this summer. I think that you will instantly recognize the problems that we are trying to solve with the project if I give you an example:

The Typical Editing Scenario: 7 Steps to Fix a Minor Typo



Imagine that you noticed a typo, such as a missing apostrophe, on your company's website. Let us say that you made the its/it's mistake. You were embarrassed by the mistake, and you wanted to fix it immediately. So you (1) logged into your content management system, (2) located the content containing the mistake, (3) clicked Edit, (4) scrolled to the typo, (5) typed in the missing apostrophe, (6) clicked Preview, and (7) clicked Publish.... Typo corrected, and in only seven steps!

In only seven steps? I think you will agree that adding a single apostrophe felt like too much work. It was so complicated that by the time you had typed it, you were four steps into a seven-step process. At step four, you may have even forgotten what typo you had set out to fix. The amount of work you did felt out of proportion with what you achieved: two minutes inserting one measly punctuation mark? The process may have made you anxious, too, both because the system's controls were so numerous and unfamiliar, and because the possibility loomed that you would screw up the company's website.

Fixing that typo didn't feel right. Let's look at the reasons behind that feeling.

Reason #1: the editing process is indirect.

You cannot act directly on the typo that is right before your eyes. Instead, you have to locate the typo in the backend, repair the typo there, and check the appearance of your change in the frontend — that is, preview your change.

A direct-manipulation interface is more appropriate. There is no reason in principle that you cannot click between the T and the S in ITS and type an apostrophe to correct a typo. The appearance of a page edited in this way should not change from the appearance of the page where you had noticed the typo, except that there should be an apostrophe where there was not one before, and a cursor flashing at the new insertion point. Inserting and deleting characters should provide immediate feedback; the resemblance between the content that the editors sees and a site visitor sees should be perfect.

Reason #2: the system is obtrusive.

You have to interact a lot with the system (logging in, navigating) before you can work on your content. The system has stolen your attention from your goal (insert apostrophe), redirecting your attention to the system and its controls, and it has made you duplicate the effort of locating the typo. Inserting an apostrophe is not an editing task, it's an editing quest.

Logging in to edit the website may be unavoidable. However, we can simplify and re-order the task so that you repair the typo first, and then you log in. Several steps can be avoided by making the editing interaction more direct (see Reason #1).

Reason #3: the system is unforgiving.

One content-management system actually destroys your undo/redo history when you preview your changes. Being a conscientious editor of the company website, you preview every change before you publish it. But if the preview reveals a mistake, you cannot undo it. Thanks a lot, content-management system!

Undo/redo is, like the seatbelts in your car, an essential piece of safety equipment in every interactive system. Like seatbelts, undo/redo won't save you from every accident, but like seatbelts, the system had better be equipped with undo/redo in working order.

A content editor should take pains to save undo/redo history either in the browser, in the backend, or both, so that you can change a web page with the confidence that you will not irreparably harm it.

Reason #4: lack of context and fidelity.

While you edit content, it is shown to you in a different shape and size than it is shown to site visitors. Even the preview may not match the view that your site visitors have. In the content editor, you may backspace over an invisible formatting character or insert an inappropriate line break that will only show up on your company's home page.

We can give the user more context and make the outcome of editing content more predictable by changing to a direct-manipulation interface.

What we're doing about it

At Pixo, we think that it is high time to take the strain and peril out of editing the web, so this past summer we began a project to do that. We call our solution an "in-line editor." It is a JavaScript that you can embed in any web page to make it editable using key commands and mouse gestures that most of us already know from using word processors.

The idea is that to edit a page, you just click where you want to edit, and the in-line editor cursor appears. You can type new text, backspace over text, insert new paragraphs and links directly. This summer one of our interns, Jake Smith, created a prototype that proved that we could realize our vision of direct web editing in a consistent way on more than one browser (Chrome, Firefox, Safari, and late versions of Internet Explorer). Now our staff are looking at the problems of integration with a content-management system. Watch our blog for the latest developments. Send an email (dyoung@pixotech.com) if you would like to get involved.

Comments

Very cool

By **Anonymous** on September 23, 2011

Wow, Dave. This is a beautiful idea, and very persuasive argument. Can't wait to see it.