The Z-Axis: Designing for the Future

Wren Lanier Wren is a designer with a passion for creating beautiful digital experiences. As a principal at Neo, she helps her clients make products that people will love to use. When she's not building internet things, Wren enjoys eating bagels, making trouble, and taking naps. She occasionally updates her blog and frequently overshares on Twitter.

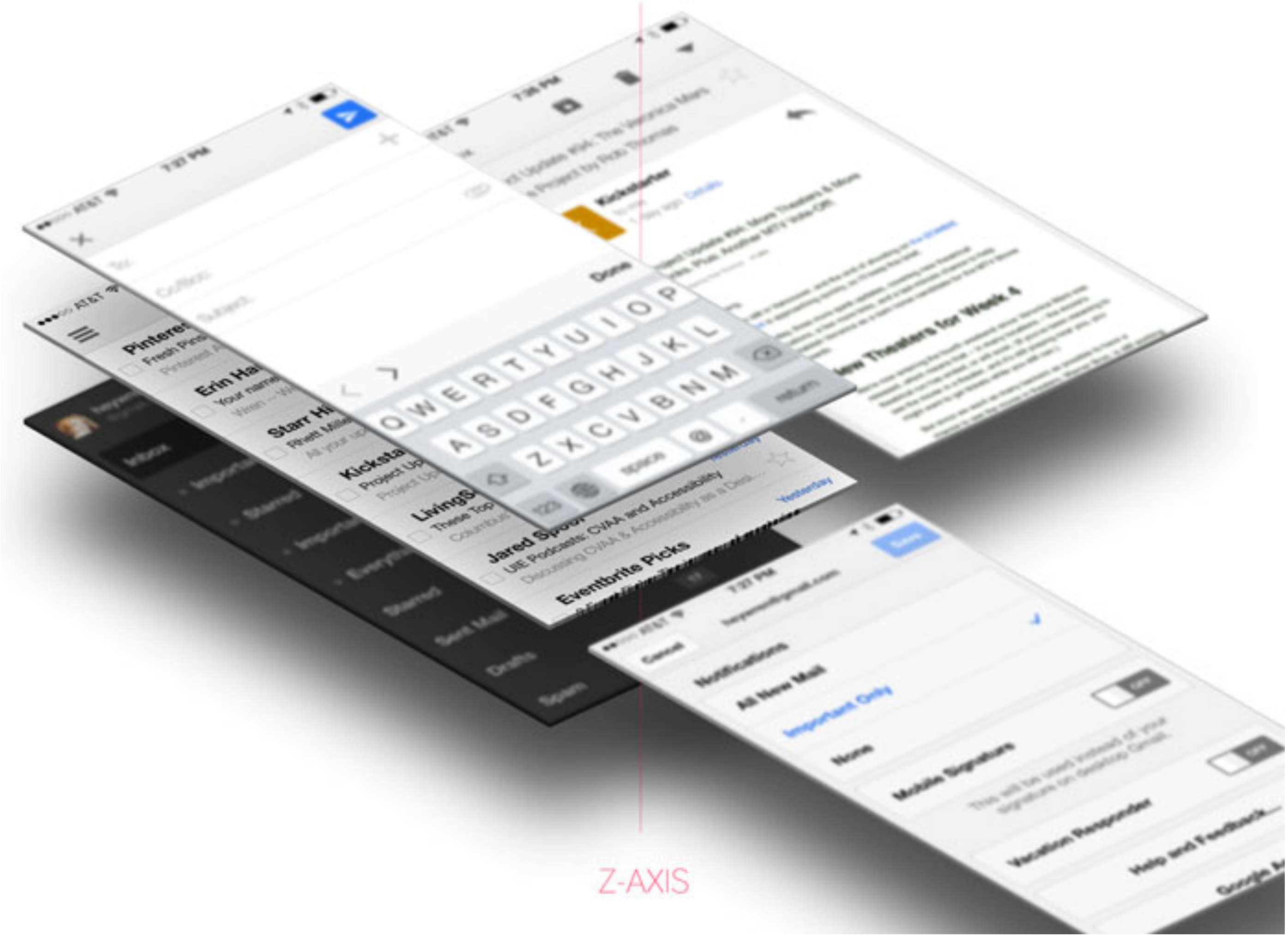
April 08, 2014

Illustration by

by Wren Lanier Published in Graphic Design

For years we've thought about the web as a two-dimensional space filled with pages that sit side by side on a flat, infinite plane. But as the devices we design for take on an increasingly diverse array of shapes and sizes, we should embrace new ways of designing up and down. By building interfaces using a system of layers, we solve tricky design problems, flexibly adapt to a variety of screens, and create new patterns that will point the way to future interactions.

In geometric terms, the z-axis is the vertex that measures space above and below the x- and y-axes. Translation for those of us who napped through geometry: it's how we describe panels and layers that sit above or below one another.



Designing on the z-axis simply means incorporating three-dimensional physics—as represented by the z-axis—into our interface designs. Not the faux-depth of skeuomorphic text shadows and button highlights, but an interface made of components that exist on distinct layers and move independently of one another.

As Andy Clarke has noted, the page is an outdated metaphor for what we're designing today. Unlike the permanence of ink on paper, a website is a series of dynamic views that can occur in many combinations. Applications require us to consider numerous happy and unhappy paths, and even static marketing sites need reusable design components that can adapt to different content needs.

By using the z-axis to place interface elements above or below one another, we can create better design systems that are more flexible and intuitive to use.

Using the z-axis to solve design problems

While juggling the constraints of making an interface work across many different screens, I often encounter the same problems. *Where do I put this? How do I make this effective for touchscreens? What can I show right away, and what can I hide?*

The answers aren't easy, but fortunately we can count on the z-axis to be there when extra pixels aren't. Got an options panel that just won't fit? Trying to add filters but the excess UI clutter doesn't seem worth it? When you're running out of space and searching for a clever solution, remembering that you have three dimensions to design in can save the day.

Creating an interface that seamlessly works across the z-axis requires two important elements: layers and transitions.

1. Layers

Incorporating layers is the key to designing on the z-axis, since layers are the way we differentiate levels above and below one another. A layer might contain a single UI element that sits above the rest of the view, or it might be a full screen that appears and disappears as necessary. However you incorporate layers, each should have a purpose—a reason it exists—and be used consistently throughout your site in a way that helps users better understand your design.

A panel that covers up the entire interface, for example, should be one of the most important functions on a site. On the other hand, an option in a secret panel that slides out from behind another object should relate to whatever sits above it, but be less important.

Menus

Generally speaking, the higher something sits on the z-axis, the more important it is. Primary navigation menus are usually placed on a higher level than other elements; they might pop over the rest of the view, they might stick to the top of the screen, or they might be accessed by zooming out to a larger menu presentation.

Teehan + Lax takes this to the extreme with the menu overlay on its website. It's more than a popover; it's like a page takeover. *Look at our menu!* it shouts. The sliding animation combined with a new screen layer grabs the user's attention, while huge font sizes and a larger-than-usual menu of links deliver more content than a typical primary nav bar and (probably) justify the need for a separate layer.

Do I love this bold menu presentation? Yes. Do I think it's a best practice we should incorporate into every site we build? No way. Not every site needs that much dramatic flair.

But I love how this inspires me to think about a menu as a piece of content in and of itself, and not just more interface cruft. Teehan + Lax highlights the act of presenting a menu to the user and how it can be more than popping up or sliding over from the left—it can be an opportunity for surprise and delight.

Action buttons

Primary action buttons, such as checking in or adding a new post, are often placed above other elements on the z-axis. It's easy to tell what an app thinks is its most important feature when it's sitting on top of everything else. Just take a look at Facebook's chat heads.

Right now, Facebook clearly thinks that messaging is its most important feature. (If you're unconvinced, Facebook also has a separate Messaging app, *and* recently paid $19 billion for What's App.) Since layers allow elements to remain fixed in one place while everything else moves around them, floating action buttons are an easy way to make them more prominent without taking up a lot of valuable screen real estate.

The z-axis gives Facebook an easy way to keep messaging front and center, and even if I don't like tapping on the disembodied faces of my friends and family, it seems to be working. For clients who want a button to "pop" a bit more, using the z-axis to give it its own layer is one of the more elegant possibilities.

Storytelling

Objects on different layers of the z-axis can move at asynchronous speeds during scrolling. This effect—usually called *parallax*—was pioneered in video games, but it's become quite popular in interactive design. When objects move at different speeds, it creates the appearance of depth and simulates the passing of time, making parallax a powerful tool for online storytelling.

Superfluous use of parallax as a trendy eye-catcher has been rightfully criticized, but the ability to move layers independently of one another allows us to animate stories on the web in a way that hasn't been as effective without the use of video. Sites like Let's Free Congress and Inception Explained use asynchronous scrolling to turn what could have been flat infographics into visual narratives. By breaking elements apart using layers, each thread can unfold at its own speed while the user controls the pace of the action.

Web designers have always worked within the confines of flat, pixel-based screens, forcing complex interactions onto two visual axes. Layers on flat screens are a hack, but an important one; they're the first step toward the true multidimensional interactions that are only a few years away. By creating layered patterns in our interfaces now, we help prepare users—and ourselves—for what's ahead.

2. Transitions

When you use layers in an interface design, it's important to include animations that smooth the transitions between them. Animated transitions serve several important functions: they help soften what could otherwise be a jarring moment of change, they describe where you came from and where you've arrived, and they provide cues about how information on the new layer relates to everything else.

Sliding

Sliding is one of the most common animated transitions because it's relatively easy to execute and simple to understand. Navigation menus, hidden panels—just slide them out quickly whenever you need them, right? But like anything "simple," sliding requires more care than you might expect.

The ubiquitous left-hand menu, used in many mobile apps including Gmail, is a perfect example. When activated, Gmail's menu doesn't slide anywhere;

it's actually the main window that slides to the right, revealing the menu on the left underneath your inbox.

The distinction is important, because the ability to see the first few words of each subject line keeps the inbox functional even when the menu is engaged; without that persistence, there's little point to the inbox remaining there at all. Mobile websites that seek to mimic this interaction should take note—sliding a left menu over the top of a webpage usually feels clunky and intrusive compared to sliding the main view over instead.

You can also slide existing elements out of view to reveal hidden panels. Tweetlist slides the keyboard down to show additional tweet options like geotagging or attaching a photo. It's a clever way to display secondary features that don't need to be visible at all times, and using the back of the keyboard reinforces the relationship between these options and sending a tweet.

Zooming

Zoom animation has been around for a while, but its frequent use in Apple's iOS 7 has increased both its popularity and its infamy. Some people have said the zooming used throughout the operating system—particularly when opening and closing apps—makes them nauseous. While this may be the case, it's worth understanding the different ways we can use zooming to transition from one layer to another, and why some types of zoom may be more stomach-churning than others.

Enlarging or shrinking single objects has been a common animation in the Apple universe since the release of Mac OS X and the introduction of the dock. It naturally found its way into the mobile world on the iPhone, and users quickly grew accustomed to tapping a photo and zooming into it to see more detail.

In the case of photos, zooming is a simple illusion created by enlarging the image. Everything around the photo remains in place; only the photo itself moves.

The zoom effect used in iOS 7 is more complex. It works by moving the "camera" in and out as you open and close apps so that everything on the screen changes, not just one object. When you close an app, for example, the app window shrinks down into its icon on the homescreen. Watch the background behind the window and you'll see all the other homescreen objects zoom back into the view as well.

This key difference—zooming the camera rather than a single element—creates a much more immersive illusion. It shifts the viewer's perspective to a new level on the z-axis. That simulated perspective-shift adds to the wow factor by introducing an element of super-realism: it mimics real-world physics, while producing an effect that would be impossible in real life. It's no wonder designers are eager to take advantage of the possibilities it offers, in spite of the potential side-effects.

This design experiment from Creative Dash shows how zooming the camera all the way out allows us to use the liminal space around a window. Our canvas is both deep and wide, and this takes advantage of both—though the extreme zoom depth would probably make quite a few users feel sick.

Foursquare has used a much more subtle version of zooming the camera to reveal map details. You don't travel very far forward, but the zoom-in reinforces the notion that you're going to a deeper level of information.

Whether you apply zoom to a single object or an entire view, it's important for the animation to be consistent with the information hierarchy you're using. When you move the camera out, you should be at a higher level, while zooming in should provide more detail.

Other transitions

Sliding and zooming are two of the most common animated transitions used today, but there are other options, including flipping or folding.

Three-dimensional objects have two (or more) sides, but most user interfaces are like the moon: they have a "light" side that's always visible and a "dark" side we never see. Flipping an object over creates a new visual space that's easy for users to understand. The only downside? Flipping is, well, *flipping slow*.

While flipping is sometimes applied to create a more magazine-like feel, 180 degrees is a big transition; it often feels slow and disruptive. In contexts where speed is critical, the time a flip adds to interactions usually isn't worth it. That said, if deployed in the right place in the right way, it could be *flipping fantastic*. Card-based layouts offer plenty of opportunities to flip objects over and reveal additional information.

What comes next

Gesture-based command centers, holographic interfaces—whatever technology lies over the horizon, we'll be better prepared to adapt our skills if we understand the principles of designing for information, not just visual tricks for laying things out with boxes and color. Just as print designers once learned to take their talents to the web, we need to learn to take our talents beyond the screen—and getting comfortable with the z-axis is a great place to start.

About the Author