

UXPin

Designing UX with Developers:

Introduction to Collaborative
Wireframing & Prototyping

UXPin

Designing UX with Developers:

Introduction to Collaborative
Wireframing & Prototyping

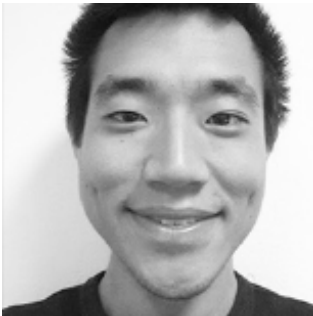
Copyright © 2015 by UXPin Inc.

All rights reserved. No part of this publication text may be uploaded or posted online without the prior written permission of the publisher.

For permission requests, write to the publisher, addressed “Attention: Permissions Request,” to hello@uxpin.com.

Index

A Few Quick Words on Wireframing & Prototyping	6
Wireframing for Developers	8
Designing for the Common Language of Interaction	14
Test, Gather Feedback, Iterate	17
Conclusion	20



Jerry Cao is a UX Content Strategist at UXPin where he gets to put his overly active imagination to paper every day. In a past life, he developed content strategies for clients at Braffon and worked in traditional advertising at DDB San Francisco. In his spare time he enjoys playing electric guitar, watching foreign horror films, and expanding his knowledge of random facts.

[Follow me on Twitter](#)



Jake is a writer and designer with a focus on interface design for the web. He often writes about W3C specs and the newest trends in web design. You can find out more on [his portfolio](#) or view his latest design work on [Dribbble](#).

A Few Quick Words on Wireframing & Prototyping

Although wireframing and prototyping are two different tasks, they both represent a web or mobile interface in its most fundamental stages.

Meant as an explorative exercise and an early visual specs document, wireframes are a natural part of the [design process](#). And as we described in the [Ultimate Guide to Prototyping](#), you can then turn a wireframe into a low-fi prototype by adding interactions or jump directly into a paper prototype.



Photo credits: [UXPin](#)

Just because the design team creates the wireframes and prototypes doesn't mean that developers have nothing to offer. In fact, developers can share early opinions while the wireframes are still rough, which helps catch small nuisances and even offer potential improvements for rough layout concepts.

In this pocket guide, we'll explore some guidelines for wireframing and prototyping so that they communicate the experience as clearly as possible to other designers, developers, and stakeholders. If you find this guide helpful, feel free to share with others.

For the love of UX design,
Jerry Cao & Jake Rocheleau

Wireframing for Developers

The best way to approach a wireframe is as a [blueprint for the product team](#).

Before adding any details, you must first lay out the headers, footers, content areas, and the relationships between these page sections. In our experience, we prefer to first wireframe the homepage (so we can start to think about content flow at the broadest level), then dive into landing pages and finally secondary pages (like About Us or Contact).

If developers are expected to code an HTML prototype straight from wireframes, then it's important to give them as much information as possible. In this case, you'll want to lean towards higher fidelity with crisp typography, richer colors, and high-resolution images since the wireframe will serve as the primary visual reference. If you prefer wireframing on paper, we recommend the "[sketching in layers](#)" technique since it provides more structure and detail than standard freehand sketching.

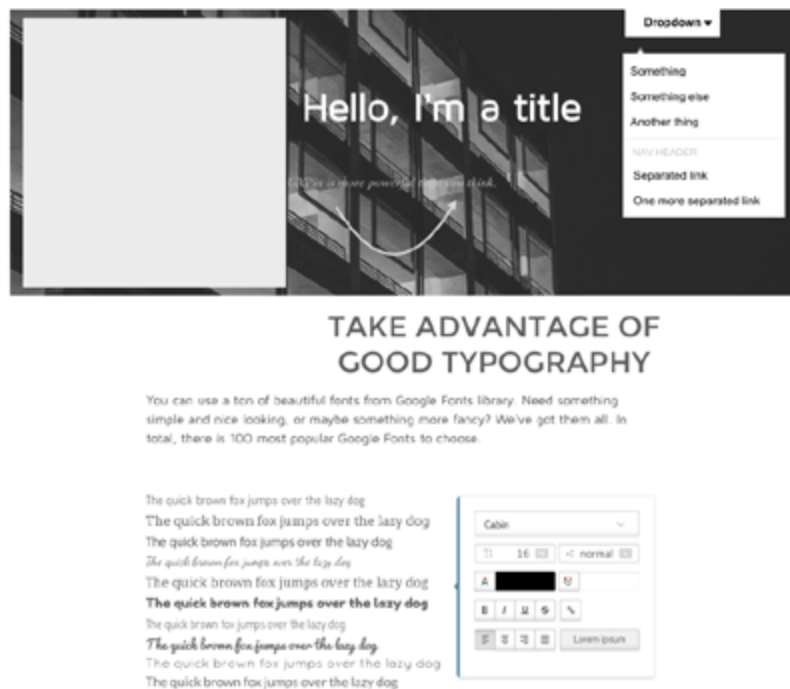


Photo credit: UXPin

On the other hand, if you're going to iterate the wireframe into a [rapid prototype](#) using a specialized tool, then the wireframe doesn't need to be as formal (although annotations are still helpful for later reference). In this situation, your prototype instead serves as a living representation of your technical specs.

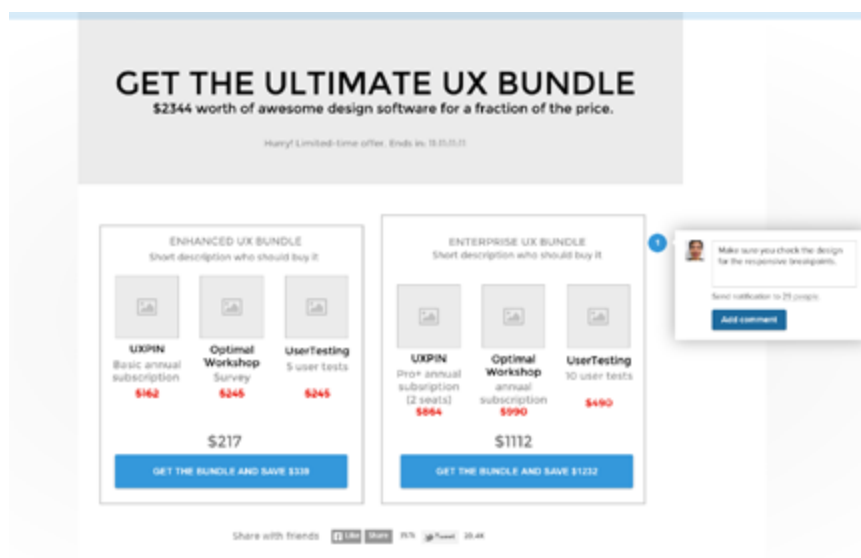


Photo credit: UXPin

When wireframing, clear explanations are vital because designers, developers, and non-technical stakeholders should be able to review your designs and quickly understand how they work. The best wireframes incorporate notes in the margins expressing which elements should be clickable, animated, or dynamic in any way.

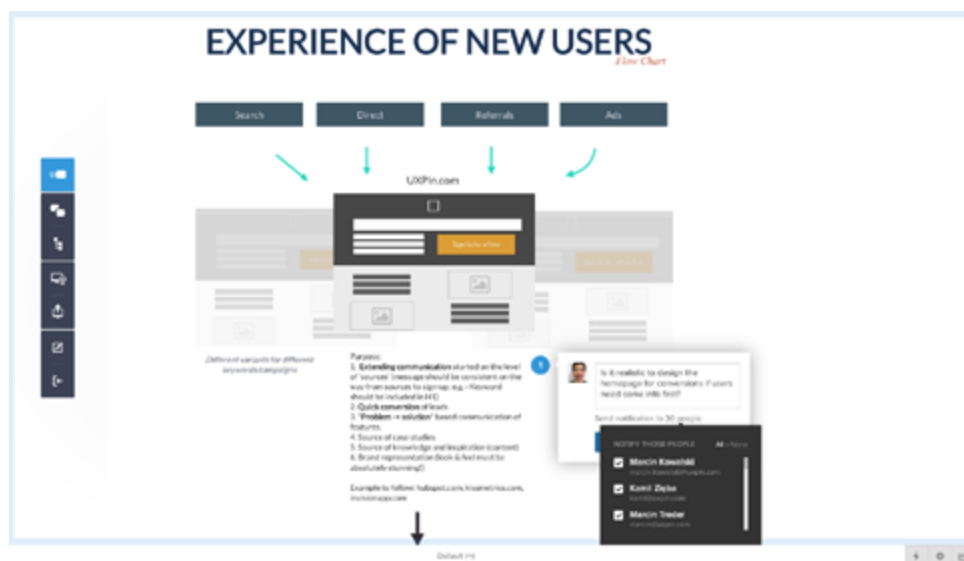


Photo credit: [UXPin](#)

But you can also create similar documents for telegraphing the intent of pages or page elements to developers. Here are a few common supplementary documents (which we first described in [The Guide to UX Design Process & Documentation](#)):

- **Storyboards** – Storyboards focus on the movement of elements and which links direct to which pages, creating a visual narrative for the entire team. Some links may not even load pages, but instead trigger effects like dropdown menus or modal windows. Created through [sketchboarding](#) or digital mediums, storyboards give developers a snapshot of the entire experience so they understand why some complex interactions might be necessary.



Photo credit: "Customer journey storyboard." [visualpun.ch](#). Creative Commons.

- Flow charts** – Flow charts are similar to storyboards except they focus more on content itself. A [website flowchart](#) usually resembles a visual sitemap, except in a more multi-directional format that is more descriptive than the traditional tree structure. Detailed flow charts can be extremely helpful for developers since you could annotate how content is served (e.g. whether a lightbox is triggered via Javascript event or AJAX). If you lack the technical knowledge, map out the site pages in the flowchart, then collaborate on the technical notes with the developer.

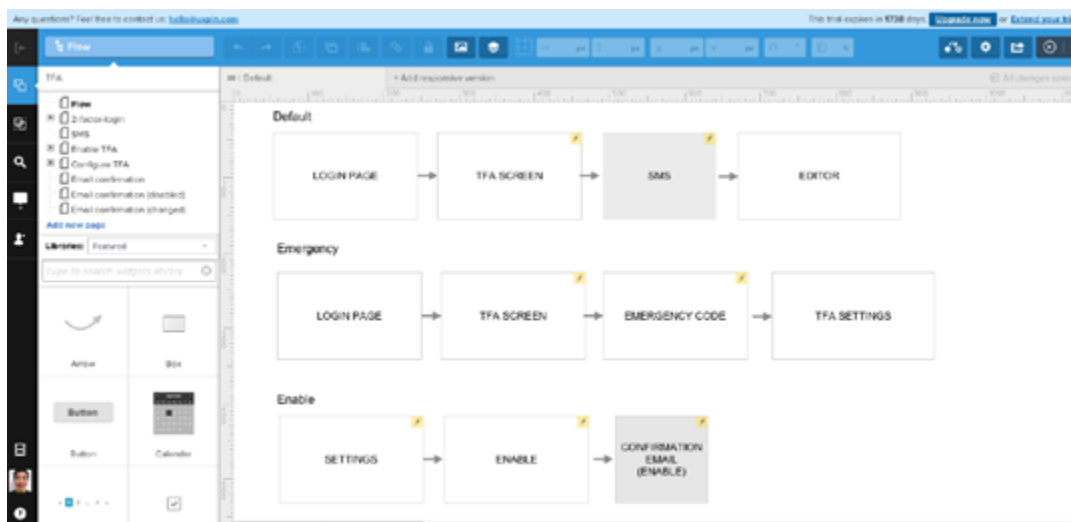



Photo credit: [UXPin](#)

- **Personas** – Like we previously discussed, the strongest way to align design and development is to focus everyone on user needs. Don't just include job titles and demographics. Dive into fears, ambitions, behaviors, goals, and habits. Create **lightweight personas**, print them out for the product team, and let them become everyone's best imaginary friend.



JONATHAN VIZZIER

"Design isn't just how it looks, it's how it works."

Demographics:

- 27 years old
- Masters in Visual Design
- Visual Designer
- Single
- Earns \$85K per year

list [text](#)

Behaviors & Beliefs:

- Obsessive over visual quality
- Hates when product managers use the word "just" before describing last-minute tasks
- Wants to be as involved in the design process as possible
- Loathes jargon, wishes people would get to the point

Characteristics & Attributes (0 to 5)

- Design experience: 3
- Education: 4
- Tech Savviness: 5
- Ambition: 5
- Workload: 5

Goals:

- To build a strong portfolio, regardless of whatever job I'm at
- To start mastering UX design by the end of this year for a career transition
- To rise up in his company and start getting assigned larger-profile projects
- Wants to help the product team see the value of emotional design, not just "core KPIs"

Photo credit: [Persona Tool](#)

While you want to keep documentation as light as possible, don't underestimate its power for updating developers as fidelity increases. In our experience, developers tend to be highly logical people who

enjoy the security of tangible visual requirements. When you treat design documentation as a collaborative tool rather than a “hand-off-and-pray-for-the-best” document, you’ll find that developers will know what to build and designers will have a quick reference for popular and unpopular ideas.

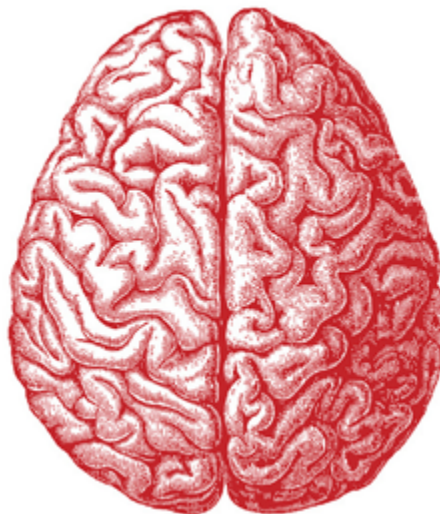
Whatever your documentation methods, make sure they complement rather than supplement your design process.

Designing for the Common Language of Interaction

When putting together digital interfaces, look at all the design features as a set of jobs. Each element on the page must help advance users toward completing their goal(s).

As described in *Interaction Design Best Practices*, it can be a good idea to think in terms of components rather than page elements. Which components on the page require action? What task does this action perform?

An interesting point to note is that developers speak the language of interaction. Developers review a wireframe/prototype and mentally formulate how it should be built. In some instances, developers might even mirror a rapid UX prototype with their own quick HTML/CSS/Javascript prototype to test the interactions.



This is the beauty of designing for interaction: it helps everybody in the creative pipeline.

When designing for interaction, you'll spend most of your time in prototypes. It's harder to wireframe interactions since you'll need plenty of annotations, but it is certainly possible (and especially helpful for mobile interactions where you want to quickly show gestures).

Wireframes typically fall into two categories: low-fidelity (gray boxes and shading) and mid-fidelity (more detailed layout with crisp typography, images, and even real copy). Prototypes span the [entire spectrum of functionality and fidelity](#), ranging from paper prototypes on the low end to life-like representations on the high end.

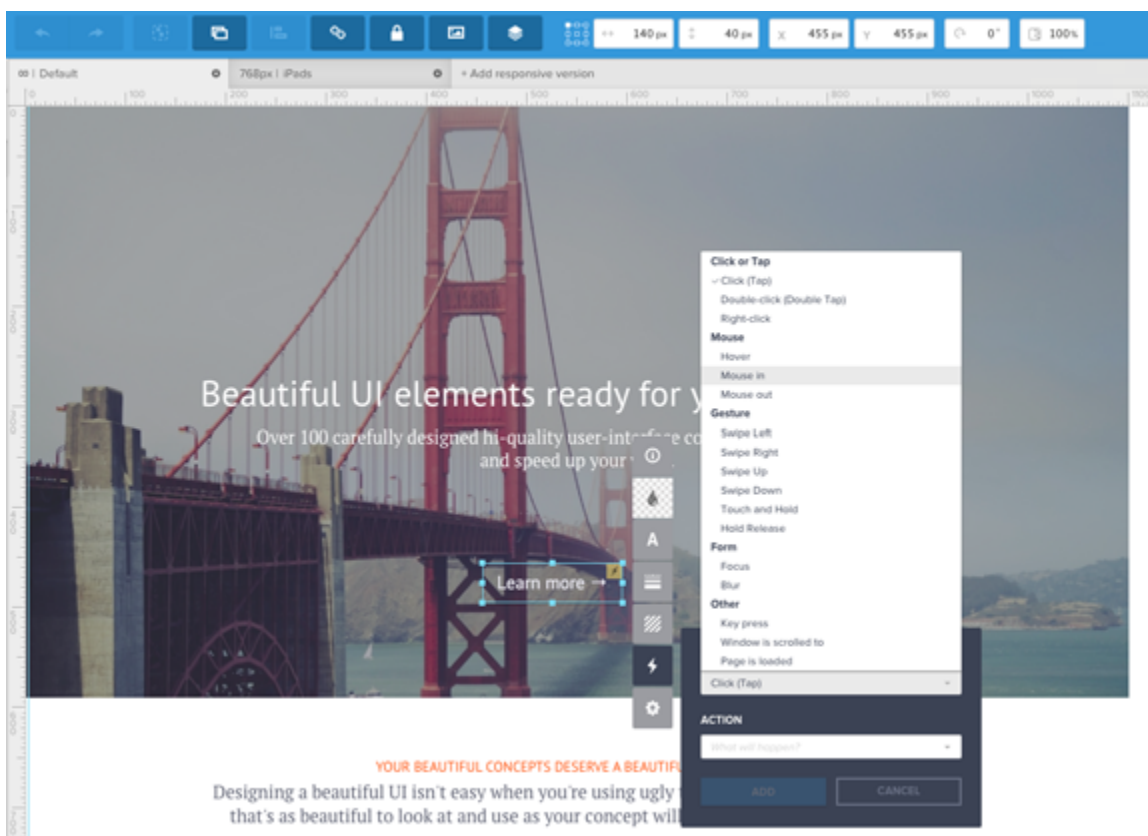


Photo credit: UXPin

Mid-fidelity wireframes are also used as a middle ground between static and interactive design and are usually less common in today's age of rapid prototyping. Mid-fi wireframes could include other documents like the storyboards we mentioned earlier. A traditional workflow starts with low-fi wireframes which are then critiqued by designers and developers and iterated into low-fi prototypes and finally hi-fi prototypes.

Sometimes designers and developers prefer to jump right into a project by moving onto prototypes from basic sketches. Truthfully, there is no “right” workflow since every team will have their own unique style.

Regardless of your process, the most important thing to remember is that you design just the right level of fidelity. Otherwise, you might jump headfirst into a hi-fi prototype (dragging your developer with you on complex discussions around animations/interactions), when all you need is a low-fi prototype to explore structure and page flow.

Be proactive and set the right expectations around the design, because stakeholder feedback also indirectly affects developers. When possible, work with the product manager or project manager to outline a project map so everyone knows what to expect regarding the number of drafts, possible revisions, and levels of fidelity for prototypes & wireframes.

Test, Gather Feedback, Iterate

Make sure you invite developers to all feedback sessions, whether it's a simple [user interview](#) a [moderated usability test](#), or an internal feedback session.

1. Conducting usability testing with developers

Not only can you both synthesize feedback in real-time, collaborative user research creates a stronger sense of joint ownership and creation. Interestingly enough, Xerox actually [sends their designers with service engineers during on-site visits](#) to deepen their understanding of how customers use the products.



Photo credit: [“Wikimedia User Testing”](#). Blue Oxen Associates. [Creative Commons](#).

Of course, collaborative usability testing isn't just about sending pairs of people into the field to observe users. As we described in

the *Guide to Usability Testing*, it can be as simple as drafting up the core tasks together, then presenting the tasks to 5-7 users and asking them to think aloud as they interact with the prototype. Study which features draw the most attention right off the bat and which receive negative reactions. Then after the test is over, ask follow up questions with your developer about where they felt most confused and experienced greatest difficulty.

2. Team Feedback & Fixes

Usability tests are meant to pinpoint issues or potential problem areas, but they don't always generate solutions. This is why teamwork and further discussion is required to flesh out the best solutions for individual problems.

Here are five UI/UX ideas to keep in mind regarding [usability 101](#):

- **Learnability** – Can people learn & adapt quickly?
- **Efficiency** – Are specific tasks easy to accomplish?
- **Memorability** – Does the interface leave a lasting impression?
- **Errors** – How are mistakes or inadvertent actions handled?
- **Satisfaction** – Does the interface provide a pleasant experience?

You can apply these questions to both the interface as a whole and to smaller features. For example, how learnable is your dropdown menu? Would it be more efficient to speed up the animation? Do users recognize that navigation bar links have dropdown menus?



Photo credits: "Question!". Stefan Baudy. [Creative Commons](#).

Rather than scrounging up ideas on your own, it's quicker to devise solutions as a team. Designers should always look toward developers for pragmatic suggestions. For example, if you're on a tight timeline, the developer may have an elegant technical solution that just needs some visual massaging.

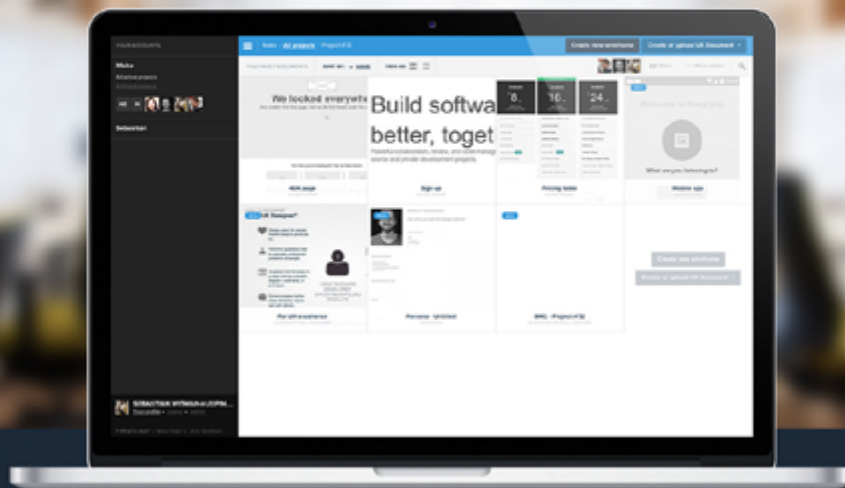
Conclusion

Early design stages like prototyping and wireframing provide fertile ground for collaboration. These methods of visualization are meant to convey how the final product looks and behaves.

Ideas flow much easier when incorporating suggestions from both designers and developers. Developers can learn a lot about design and what makes the interface work. Developers who sit in on usability meetings will also absorb new ideas that later improve their ideas in the development stage.

When it comes to the UX design process, everyone is responsible for keeping the team involved and moving forward.

[Start wireframing & prototyping in with UXPin free trial \(7-30\)](#)



- ✓ Complete prototyping framework for web, mobile, and wearables
- ✓ Collaboration and feedback for any team size
 - ✓ Lo-fi to hi-fi design in a single tool
- ✓ Integration with Photoshop and Sketch

UXPin

www.uxpin.com