webdesign.tutsplus.com

The Basics of Great UX

by Justin Smith • 7 min read • original

To the benefit of software and applications everywhere, UX has become an increasingly important step in the Software Development lifecycle. Even though UX is a slightly surreptitious layer of design, it's no less important. It, by the nature of its name, defines the experience of the users. That experience determines whether or not they want to come back for more, or run screaming in the other direction.

UX is something anyone can do. The problem is not everyone can do it well. I can undertake the visual design for a website, but I guarantee you that it will look dreadful and unprofessional.

If you learn nothing else about UX I ask you to remember these two things.

1. Know thy user.
2. You are not the user (in most cases).

If you don't know your user base, their habits and tendencies, and why they do or do not perform certain actions, then you can't be expected to design a good experience for them. Even if you are very similar to your users, remember that you are only one person and that does not define the qualities of your user base as a whole. Get to know your users and design the experience with them in mind. The role of a UXer is to have that knowledge, constantly expand it, and then design with those users in mind, crafting amazing experiences that will delight and fulfill them.

Let's look at all the possible steps a UXer might take in defining the UX of a product. These steps create the ideal process. These steps aren't always possible to complete in the real world, but we need to cover all of them so that you're aware when you can leave out certain steps and why. Sometimes you don't necessarily leave out a step, but it is melded into another step, or replaced using a combination of experience, knowledge, and intuition.

Requirements

Requirements Gathering is one of the first steps in designing the UX. During this stage you need to ask a lot of questions. Many questions may not be able to be answered right away but note those and be persistent. There are several types of requirements:

1. **Business Requirements**: The goals and needs of other parts of your company or what's necessary to monetize the product. Unfortunately, this often trumps some things you may want to do. It's a necessary evil if your product is anything beyond a project for pure enjoyment.
2. **Design Requirements**: Sometimes there may be special design considerations or needs that must be met.
3. **Technology Requirements**: There could be a specific technology need (platform, language, etc.) you need to consider in the design. What are your limitations?
4. **User Requirements**: Who is this product for? Who is the main audience? Is there a fringe audience and, if so, who is it? Does it cover your entire user base or support a subset?

Block building on Photodune

These requirements will be the foundation that will shape your design. Without these your designs will be aimless.

User Analysis

Before we can design for a user we need to know our users. Sometimes you'll have a lot of preexisting knowledge about them. Sometimes you'll only know a handful of assumptions about them. If you don't have knowledge of your users then your design is an educated guess at best. User Analysis is necessary to understand the needs and propensities of the users.

Little users on Photodune

You'll need to ask the following questions at minimum to get a grasp on your user base:

1. **Who is this for?** Demographics? Likes/Dislikes? Hobbies? Occupation?
2. What **special requirements** do they represent? Do they present a particular problem you need to solve? Do they fit a particular business need?
3. What **mental models** of the user base do you need to consider? Do they vary significantly within subsets?
4. **When, Why, How** would the users use this product?
5. What **accessibility concerns** do you need to include in the design?

Task Analysis

After we've completed our user analysis we need to move onto the task analysis. **What is the primary action the users need to perform?** There may be many things someone can do on your site or application but there's always a main task. You need to figure out what that task is and optimize the UX for that task for the particular user base for which it is purposed. We still haven't reached the designing stage, but we now know what we'll be designing, and for whom we're designing it.

Reminder notes on Photodune

We also need to determine any secondary tasks. There are very few sites/applications where users can only do one thing; there are often several related tasks to be performed. Scope out all of these to know the breadth of what you're designing toward.

Other things to figure out in the task analysis:

- What kind of Help/FAQ do you need?

- What error states are needed?
- What are the fringe cases your users might present?
- Are there multiple methods your users might try to perform to accomplish a singular task?

Function Allocation

Function Allocation is figuring out what needs to handle all of the functions you've determined need to happen. This occurs on several levels.

- Do you have **multiple systems and/or servers**? Which ones will handle a function the best.
- What needs to happen on the **back-end vs. the front-end**?
- Which **pages** are best suited to handle a particular function? (Card sorting works great here.)
- What is **automated** (handled by the computer) vs. what is **manual** (controlled/handled by the user)?

Fork, spoon and knife on Photodune

The answers to these questions can dramatically influence the efficiency and usability of your product.

Sketching

Sketch. A lot. It's quick. It's cheap. It's effective.

Sketching gets a lot of ideas out of your head and onto paper quickly. It helps shape good ideas and eliminate bad ones. You can iterate on sketches very quickly. The investment is low for the return.

Sketching house on Photodune

Creativity is also key in this step. Do some wacky, off-the-wall things. They may not work but they could spur another idea. Try to vary your sketches significantly to see how far you can stretch an idea.

Pen/Pencil + Paper = A bank to draw from for when you start to really define your designs.

Mid-level Wireframes

Now you should have at least an idea of where you want your design to go. The first set of wireframes should be an iteration on your sketches and start to bring out more definition. The information architecture should start to take place. This is where things go from a bunch of loosely connected concepts and pieces of information into a more cohesive documentation of how the product looks and functions.

Prototypes

There are lots of different prototypes you can do. These can range from a very simple clickable PDF to an almost fully functioning HTML/CSS website. It really depends on what you need for your project as to how far you go with it. There are plenty of tools out there though to get your prototypes up very quickly. A small sample includes:

Figure out the tool that suits your workflow and knowledge base to figure out what you can prototype in quickly and effectively. Prototypes shouldn't take a long time, but they should effectively communicate your design and interactions.

Prototypes help figure out where any weirdness lies in the interaction and what it feels like when the product is actually in motion. It can help uncover some things static wireframes may not be able to fully uncover.

High-fidelity Wireframes

This is where you then go back and start to make the wireframes high-fidelity. Add all the definition you need. Cover as many details about the layout and interaction as you can. Anything you leave open to

interpretation may easily be taken the opposite way you wanted. Don't assume anything.

Better visual quality may or may not be necessary. Most of the time this really should be taken care of in the visual design phase. I've found it's often better to fall back on more wireframe-y visuals in order to avoid confusion with final colors and visual design specs. However, I do use color if its related to functionality (e.g. red for error messages).

Usability Testing

This step can take place wherever you need it to during the design process. It sometimes fits at the end and sometimes more towards the beginning of a project. However, this is where you get actual user feedback. They may affirm your suspicions. They may turn your thoughts upside down. This is crucial though to know their thoughts and where their problems lie.

Senior man using computer on Photodune

You are designing for the user. This feedback should drive your design or redesign in order to address the issues that arise during this testing. Discounting or ignoring this feedback, sans true outliers, is unwise and an absolute UX sin.

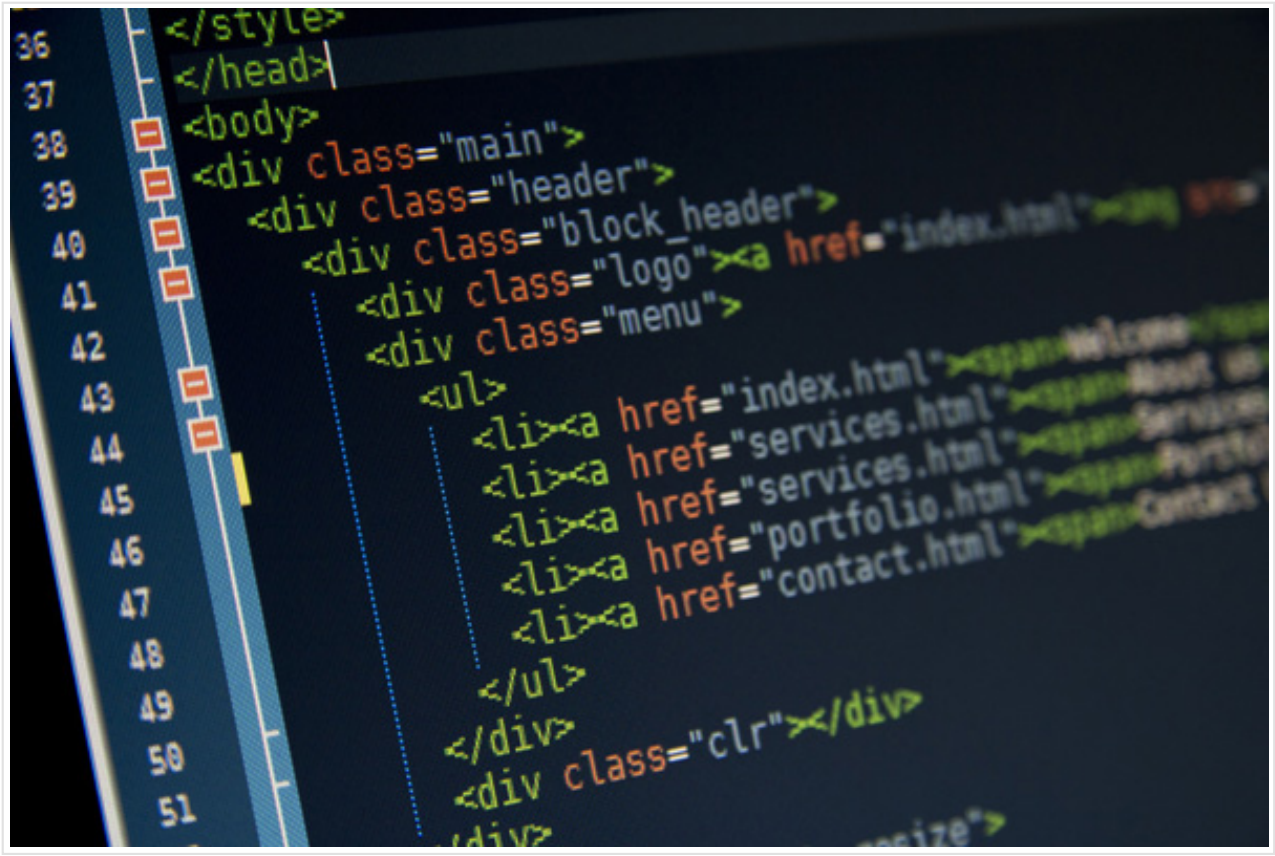
Visual Design

I've seen a lot of UX processes consider this the point in which they hand their documentation over the fence, cross their fingers, and hope for the best. That shouldn't be the case! I work very collaboratively even after I'm done with the UX proper.

Your visual designers should also be thinking about the UX. If they're not I would contend they're not fully doing their job. Making your wireframes look better is great but if they have an idea that makes the UX even better be willing to listen and incorporate. Often designers think different enough to spur great ideas and improvements.

Development

Similarly, you should be working with developers. This may not be quite as collaborative but there are often issues that arise during this stage in which you'll need to adjust the UX. This also helps you to gain an understanding of the technical limitations for future projects. Your developers will be much happier for you to gain this knowledge rather than designing impossible solutions.



HTML Code on Photodune

Also, test your UX against the final product. This is where you go back to the wireframes and make sure what's going out to your customers is what you designed. It's not uncommon that something doesn't match up and you should be comfortable enough with your developers to point those issues out. Ultimately, you are responsible for the experience of your users, so make sure they're getting the product that you designed.

Conclusion

These are just the basics to putting together great UX. You may not be able to perform all of these steps, but you need to be aware of them all to know when you can leave them out.

I personally believe UX is not a formula. It differs project-by-project and relies on an amalgamation of knowledge, experience, research, and intuition. We'll dive further into specific areas in future articles, but we

first needed a basis of what a UXer does on any given day at work. The steps outlined above will get you on your way to knowing how to put together great UX for products your users will love.

Original URL:
http://webdesign.tutsplus.com/articles/the-basics-of-great-ux--webdesign-8823