Web Design Handbook

V1 - By Kyreena Hay

Contents

1. Introduction

The purpose of this document is to act as an informal, comprehensive outline of the considerations at every stage of the web design process. I am hopeful that we can use and expand upon this as a reference point that can help us all build upon our existing knowledge so that we can all grow together as a team - and of course build awesome web products for our clients!

2. General:

a. Progressive enhancement is an important web design principle where design is considered for the content first. In practical terms, we design for the most feature-restricted scenarios first, adding on additional layers for those devices and scenarios with wider support for better features. For example you might want a stickied navigation bar, but if your target audience largely uses devices that don't support them, then you will need to forego a stickied navigation in favour of an approach with wider support.

b. It's very important to use fonts that are optimised for web (web hinted), or you

will have issues with font-rendering. Windows in particular renders fonts with little regard for curve precision, so many fonts are specifically optimised to prevent this issue and maintain legibility. You will also need to consider a fallback font-stack that is found on all computers eg. Arial, Times, etc, in case your custom font doesn't load for whatever reason. Additionally, where your audience is located will matter and you will want to host your custom fonts in the same locale to reduce load delay. Note that licensing is also very important (as not to open the company up to lawsuits) so ensure that typefaces used have an appropriate, DRM-free license attached (DRM is bad in NZ).

c. Draw wireframes. I recommend against doing them digitally, as you become attached to them in that more polished format, which reduces the effectiveness of wireframing - which is to explore the usability, flow, and functionality of the website iteratively.

d. Once you're happy with the use-flow of the website, the general layout of elements, and have sign-off from the client, you move on to the more detailed wireframes, or mockups. These require content, and are again iteratively designed, through exploring many concepts.

e. Once a final mockup is chosen and signed off, you move on to the final design stage, which is where you flesh out the full responsive look, then sign-off before handing off all deliverables to the devs. Design absolutely needs to be finalised before it's handed to devs, as design changes after this stage require very costly and time consuming code revisions.

f. Max website size should be 990px, and a grid should be used. The best, easily divisible for responsive grid is 12 columns with 20px gutters, and 20px margins. In Photoshop, you can use the GuideGuide plugin to generate accurate grids - well, as accurate as is possible anyway; don't be alarmed when one column isn't quite as wide as the others, as this is the limitation of pixel rounding.

g. Resolution independence is important on the web, so utilise vector layers and high resolution smart objects where possible (In Photoshop).

h. Use layer comps for different page layouts where possible (In Photoshop).

i. Don't design in illustrator if you can help it. Fireworks and Photoshop are better - Photoshop being best, as Fireworks will be discontinued soon sadly. There are a

number of reasons for this, from the pixel rendering in Ai being less accurate than PS, to PS being better equipped for website design, as well as more accessible for developers who may need to jump in and grab colour hex values, images etc. Though really, developers shouldn't ever have to do any of that **(see point 10)**.

j. Never forget the importance of the content "above the fold"; this is where users first land, and your main priority for getting the user to stay on the page/ website. Get them interested, let them know that this is where they want to be by appealing to their needs - this is generally considered to mean providing them with information on the site, so they know they're in the right place for what they're after.

3. Client expectations

a. As a web designer you have a responsibility not only to fulfill client requests, but to go above and beyond their expectations and produce a highly successful website for your client. Web results & effectiveness are directly measurable & no client will be satisfied if their website does not perform well. Thus, it is important to realise that clients aren't trained in web design - they will not often know what will bring them the most success and often do not have their *real* best interests at heart. With this in mind, consider:

i. The single biggest stakeholder in the success of any website is the Users, by a very large margin. If you can teach clients that sound usability practice will lead them to success, then they will trust you to execute this effectively.

ii. Factors like user flow, accessibility and especially load times (see progressive enhancement, image optimisation) have been shown again and again to directly influence bounce rate/ conversions and thus are strongly correlated with resulting revenue and profit margins. Emphasise this to your client and sound usability practices should be an easy sell!

4. Responsive Design:

a. Responsive design is a progressive enhancement concept. In practical terms - the mobile design will lack features that the tablet and desktop have, as the use case and technological requirements are different. Mobile may even gain functionality or elements to make up for the loss of larger elements; it's all a part of the usability considerations important for responsive web design.

b. Designing how the site looks on mobile and tablet is imperative, and in some cases it may be necessary to include desktop view as well, though desktop and tablet are often very similar. Some good approximations: Mobile width when designing is 480px (600-800px for 'above the fold'), tablet width is 768px, and desktop is 990px+. Try not to focus too much on devices as our websites need to be future-proofed against any possible new device than may emerge in a rapidly, ever diversifying market.

5. Image optimisation:

a. Compression for web (72dpi, consideration of formats, and ratio of compression to image quality), and ensuring the image size is exactly the size it will display at in web. Images should be saved exactly at the size required for the website.

b. Consideration for progressive enhancement, balanced with the needs of retina screens, and the current limitations that CMS's have for serving up images. It's important to keep an eye on the technology, because as soon as a solution for CMS image display is found, we should use it. Right now, the best solution I have is saving images at 1.5x size (retina requires 2x, but 1.5x is a good compromise). But, when there is no CMS involved, you will need to save out a different sized image for each break-point.

6. Accessibility:

a. Consideration for disabilities, by default, is important. Color blindness, and complete blindness are especially common. For design, this means design that has high contrast (i.e. black text on white is high contrast design, while light grey text on white is low contrast and bad), body text preferably 12pt+, usability that is so simple your grandmother could use it, and copy that isn't excessive and is laid out in an easily scannable manner. For blind users your user flow must be accessible to screen-readers.

b. Some companies - like government organisations - require a full stack, unique consideration for disabilities, so if the brief requires it, often times you have to design in such a way that people with disabilities can make changes to the website to toggle things according to their needs. See: government accessibility guidelines.

7. Usability:

a. It is important to remember that while you may think a layout makes sense to you, it might not for others (especially people of different cultures or where english isn't the first language, eg right-to-left languages), which means they will

become frustrated trying to find what they want. It's best to observe common use patterns on similar websites and interfaces, and adhere to common layout patterns, after establishing the information hierarchy and designing for common use cases. This often means the creativity of the site suffers, as you must stick to very simple, common patterns, but it is what needs to be done.

b. Affordance is when something has a visual cue that indicates how you interact with it. Like a door that has a handle should be pulled, not pushed. If you have to push that door, you become frustrated because the affordance indicator suggests to pull it, which you naturally attempt first. In UI, an example is a button that looks like a button - if it doesn't look like a button, the user won't know to click it.

c. As few clicks as possible - Very important. This is a particularly good rule of thumb when thinking about navigation layers. I try really hard to not have more than one layer of navigation, and find other ways to have sub-navigation to prevent too much clicking. More than one layer of navigation requires breadcrumbs. Avoid more than 2 layers at all costs - this usually indicates something has gone wrong at the wireframing stage and represents an overcomplicated user flow.

d. For websites that have generated content and pages, the devs will tell you they need a 404 error page design. 404 is what you get when an error occurred locating the page they wanted, so that one needs clear instructions for finding the page, or going elsewhere on the site. Github has the best example of a 404 page done right.

e. Icons are a complex issue and need to be treated with delicacy. It's best to accompany an icon with text, as there are very few that all users easily recognise; we're still largely in the "training" phase of making all users familiar with iconography on the web, and it doesn't help that we're not all using the same icons for the same functions either. The Hamburger icon is a good example of an icon being abused without consideration for the user, and it's looking like there's now a movement away from just assuming you can use it to represent a "menu" (does Joe Blogg user even know what a "menu" is, in the context of web? We're assuming too much with this interface element). In reference to what was said before about "designing for your grandmother"; my grandmother would not make use of that icon (there was a study on it actually).

f. Mobile has a lot of considerations as well:

    i. Using your fingers means you need large buttons and spaces for scrolling -

you don't have the accuracy of a mouse cursor. I once designed a mobile app that had both vertical and horizontal scroll functions, and you really needed a lot of space to put your finger and swipe in the specific direction you wanted, or the device became confused and swiped the wrong way.

ii. You need to consider the use-case scenario for viewing a website on your mobile phone vs tablet vs desktop. The reading experience on mobile is less comfortable than it is on tablet or desktop, so you want to reduce scrolling and reading on mobile phones; I recommend a quick "back to top" solution for all devices.

8. User Experience:

a. It's good to consider animations and transitions for the website, as it creates an entertaining, unique, personable, and more real experience for the user. When something moves like it's affected by external factors such as gravity, it makes an otherwise robotic interaction more real for users. Web technology has advanced a lot, and there are many, many possibilities these days. I recommend checking out Codrops for a plethora of lovely animations and transitions that most frontend devs can implement with ease. Many things can be animated, from drop-downs, to buttons, and interesting elements. Keep in mind that an alternative state to hover needs to be included (progressive enhancement again), for devices that do not have hover capabilities i.e. tablets.

b. All objects that a user interacts with in some way, like a button, link, drop-down, or form, needs a state for every stage of interaction: hover, active, focus (when you select a field in a form to fill out), and disabled. You will need to design each state, where needed. This is again where you keep affordance and use of animation in mind e.g. a button pushes down when you click it, a lot like a real life button.

c. You will also need to design user feedback - positive affirmation being an important one (which is when you tell a user when something was successfully done e.g. email sent). Examples are: error messages; success message; loader when something is being processed, like an email being sent, progress bar, and more.

d. Ensure a smooth user flow at all times. Things like login, sign-up, and payment processes commonly trip up on this. Consider that the user wants as few steps and things to fill out as possible, wants to be moved to as few screens as possible, not be taken to a third party site if possible, and return back to what they were doing before without issue. This is largely fleshed out in wireframes, but there will

be design elements and compliment the process.

e. Design for states other than the standard. What happens if there is no content, on a page where content is generated (i.e. an empty cart) - do you have a call to action to rectify this? What happens if there's an error? What happens if they lose internet connection? What happens if they start a process, get interrupted, and return hours later having completely forgotten what they were doing? What happens when generated content is loading? How does that generated content look when it loads?

f. Web users are largely scanners first, readers second (they'll scan to find what they want, and read more if they're interested) - often they don't progress on from scanning. Make sure you don't lose your audience in a wall of text, and format text using principles of strong information hierarchy (ie: use strong headings, lists, and other highly scannable content formats to direct user attention rapidly through the page).

9. Technology:

a. You will need to understand and consider the limitations of the technology that will be used. Here are a few examples:

i. Internet Explorer can't do a lot of things that other browsers can, so you'll need to weigh up how important it is that Internet Explorer users don't miss out on designed features because the browser can't make it look the way you designed. This depends on your user base, and how important it is that the feature looks like it does - maybe you could have an image fallback in the code, or maybe you will just forego that design. A good example is rounded corners; IE7 doesn't support those, so we either decide to let IE7 suffer and have no rounded corners, completely forego rounded corners, or IE7 gets an image fall-back which isn't very flexible code-wise.

ii. There are a LOT of mobile browsers out there; kindle has a browser, which is fairly limited, older versions of opera mini are a relative write-off, the default android browser is a shocker, whilst safari, opera, and chrome all have their own limitations on mobile. Keep in mind your target user base and what they'll likely be using.

iii. Content Management Systems have their own limitations, in particular, a good rule-of-thumb is to create "components/ modules" - sections of

design that are cloneable and re-usable across the site.

iv. While your design may look beautiful on your mac screen, most users have a cheap monitor that isn't calibrated, so it may look bad on theirs, or faded out (lack of decent contrast). What saves this is upping contrast, not using colours that are very close to white for important elements.

v. Moodle again has limitations - largely there seems to be limited responsive capabilities. Same goes for online courses. If it can't be made to adapt to a smaller screen, a different solution that won't impede on the mobile experience needs to be found.

vi. Touch devices have no hover, as well as different common UI patterns i.e. Android doesn't traditionally use the "hamburger icon" or the subsequent drawer menu functionality, they instead use a drop-down menu aka a "pop-over". The hamburger icon is currently under great debate, and your mobile menu design decision needs to be well considered for usability.

10. Preparing the design to be handed to Devs:

a. The majority of Backend Devs have little design training - which usually means they won't have your same level of attention to detail. This can range from not seeing that you used a sans-serif font (not a serif!), to not understanding the importance of that 10px space between two elements and setting it to 20px instead. Your best bet is to simply assume the worst, and hand them all images and icons exactly as they should be on the site, give them a .txt file with all hex colour values and fonts used, and sending them screenshots with grids visible, so they see how things line up. Additionally, give them the raw .psd file so they can get exact measurements of things.

b. Name files according to the Object Orientated convention for easy and transferable storage and use. For example, that arrow that points right, will be called "icon_arrow_right.png". All icons will start with "icon", and all arrow variants will be suffixed with a descriptor. Make sure like objects can be grouped alphabetically by name in the file system.

c. Best case scenario would be if you could create image spritesheets. This is when you save the same type of images into one file, all nicely ordered and aligned. Then, just one image loads, which reduces site load time, and in the code we reference one image and just change the background position. This will however require good

grasp of code due to some very complex positioning cases (leaving space for related content) - I'll probably just end up making these myself.

d. Show devs your planned use-flow. They need to see how you intend the user to navigate the website.

e. An optimum scenario that we don't always get time for, is to give devs "functional specs", which is a matrix table of every interactive object on the site, and what its hover, active, and focus states are, what happens when you interact with them etc.

11. Some useful links:

a. Smashing Magazine: http://www.smashingmagazine.com/ (If you're going to read any of these, Smashing Mag is the one; it is the best.)
b. Responsive Resources (most comprehensive list on the web): http://bradfrost.github.io/this-is-responsive/resources.html
c. A List Apart: http://alistapart.com/
d. Excellent resource for high quality web design patterns and examples: http://tympanus.net/codrops/
e. A fairly important resource for design pattern inspiration: http://patterntap.com/
f. Great icon search engine: https://www.iconfinder.com/
g. A more general resource for everything you could possibly need: http://www.agiledesigners.com/
h. The following are great sources of web design inspiration*:
    i. http://www.siteinspire.com/
    ii. http://httpster.net/
    iii. http://www.designmadeingermany.de/sites-we-like/
    iv. http://www.verynicesites.com/
    v. http://www.minimalsites.com/
    vi. http://designspiration.net/ (all types of design here)
i. A library of UI and interaction patterns: http://useyourinterface.com/
j. These guys are the leaders in UX research: http://www.nngroup.com/
k. Excellent articles discussing aspects of web: http://24ways.org/

*Many of these are self-indulgent and don't have much regard for end-users; it's always good to find out how feasible things are - every website has a different user base and tech consideration, so what works for one, may not work for another.

A Web Design Checklist:

- ❑ Fonts are web optimised.
- ❑ Where your fonts are sourced from (and therefore hosted) has been considered.
- ❑ Your working file is laid out with mobile, tablet, and web, lain side-by-side, with the appropriate dimensions.
- ❑ Images in your file are imported at their largest possible size, turned into smart objects, then the smart object downsized according to your needs, so that they can ultimately be saved out at least 1.5x their viewed size.
- ❑ A grid is being used.
- ❑ Principles of Progressive Enhancement, and graceful degradation, are kept in mind.
- ❑ Technological limitations are kept in mind.
- ❑ Accessibility has been considered.
- ❑ The usability of icon choices and treatment have been considered.
- ❑ All interactive elements have the necessary states, like:
    - ❑ hover
    - ❑ active
    - ❑ focus
    - ❑ disabled
- ❑ Alternate states are considered eg:
    - ❑ No content
    - ❑ Search returned nothing
- ❑ All necessary user feedback has been designed:
    - ❑ Error messages
    - ❑ 404 page that makes sense and can easily direct you to where you want to be.
    - ❑ Success feedback
    - ❑ Processing/ loading feedback
- ❑ All images have been optimised for web, and labelled so that images that fall within the same category are group together through naming convention.
- ❑ You've made it easy for devs to be able to obtain hex values and sizes, either by handing over a raw file, or a document outlining all colour values and sizes.