Who is this whitepaper for?

This whitepaper is for any developer who wants to learn more about user interface design. Maybe you are a solo developer who needs some pointers on how to get started with the visual design of your app. Or you are one of the coders on a small team and want to better understand the lingo and the concerns of the designers in your group. Or you are a developer moving into a supervisory position and now need to learn how to better communicate with stakeholders, creatives and coders about the look and feel of your apps.

This document will provide a foothold into what sometimes feels like an alien world—the world of creative design, with the goal of making the unfamiliar understandable.

HOW IS THIS WHITEPAPER DESIGNED?

A document which proposes to explain design to others should naturally have for itself a design behind its expositional structure. This Developer’s Guide happens to have just such an organizing principle: a geometrical one. As with Euclid’s Elements, this guide starts with the raw building blocks of its subject matter. In user interface design, these elementary topics are:

- Typography
- Iconography
- Color

After establishing the building blocks, it makes sense to show how we combine these elements of design to create more complex structures. The subsequent sections cover the following compound topics:

- Designing for accessibility
- Communicating with your users
- Visual communication and visual cues
- Creating intuitive navigation
- Polishing your design with animations and audio cues

Finally, having covered the scientific aspects of creative design, it is important to provide some introductory remarks concerning the artistic side of UI design in the remaining two sections:

- Design thinking and inspiration
- Sketching

Although they are at the end, these final two topics are perhaps the most important for understanding why we must all strive to create good user interface design.
## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>4</td>
</tr>
<tr>
<td>Understand your customers</td>
<td></td>
</tr>
<tr>
<td>Prototypes and Validation</td>
<td></td>
</tr>
<tr>
<td><strong>Typography and Fonts</strong></td>
<td>8</td>
</tr>
<tr>
<td>Understanding typography and UX design</td>
<td></td>
</tr>
<tr>
<td>Kerning and leading</td>
<td></td>
</tr>
<tr>
<td>Fonts (and font families)</td>
<td></td>
</tr>
<tr>
<td><strong>Iconography</strong></td>
<td>13</td>
</tr>
<tr>
<td>Using iconography to enhance UX design</td>
<td></td>
</tr>
<tr>
<td>When to use custom iconography</td>
<td></td>
</tr>
<tr>
<td>Where to find icons</td>
<td></td>
</tr>
<tr>
<td><strong>Color</strong></td>
<td>16</td>
</tr>
<tr>
<td>Color theory</td>
<td></td>
</tr>
<tr>
<td>Color contrast, accessibility and schemes</td>
<td></td>
</tr>
<tr>
<td><strong>Accessibility</strong></td>
<td>20</td>
</tr>
<tr>
<td>Be accessible</td>
<td></td>
</tr>
<tr>
<td>Scale</td>
<td></td>
</tr>
<tr>
<td><strong>User Communication</strong></td>
<td>28</td>
</tr>
<tr>
<td>Friendliness</td>
<td></td>
</tr>
<tr>
<td>Provide a helping hand</td>
<td></td>
</tr>
<tr>
<td>Tone</td>
<td></td>
</tr>
<tr>
<td><strong>Visual Communication</strong></td>
<td>33</td>
</tr>
<tr>
<td>Affordances</td>
<td></td>
</tr>
<tr>
<td>Calls to action</td>
<td></td>
</tr>
<tr>
<td><strong>Designing for Intuitive Navigation</strong></td>
<td>38</td>
</tr>
<tr>
<td>Consistency</td>
<td></td>
</tr>
<tr>
<td>Rules of Thumb</td>
<td></td>
</tr>
<tr>
<td><strong>Animation and Audio Cues</strong></td>
<td>43</td>
</tr>
<tr>
<td>Animations</td>
<td></td>
</tr>
<tr>
<td>Sound effects and best practices</td>
<td></td>
</tr>
<tr>
<td><strong>Design Thinking and Inspiration</strong></td>
<td>48</td>
</tr>
<tr>
<td>Find and feed your inspiration</td>
<td></td>
</tr>
<tr>
<td>Sketching and the philosophy behind it</td>
<td></td>
</tr>
<tr>
<td><strong>Conclusion</strong></td>
<td>54</td>
</tr>
<tr>
<td><strong>Index</strong></td>
<td>55</td>
</tr>
</tbody>
</table>
Introduction

There is a well-known gap in the software industry between developers and designers, coders and creatives, that you don’t really see in other areas of the field. To improve the flow between development and QA teams, developers long ago rolled up their sleeves and came up with automated unit testing frameworks and continuous integration platforms. To better understand and improve project management, developers taught themselves how to analyze business requirements and adopted agile methodologies (lots and lots of them!).

When it comes to design, however, developers up to now have tended to run for the hills. Why is that?

DEMYSTIFYING DESIGN

Like development, design is really just made up of a core set of skills, best practices, and parcels of information. Through this whitepaper, you will gain basic mastery of these design skills.

Competence with these core skills contributes to what designers rather loosely define as “good taste”—the ability to quickly judge what works visually and what does not. By applying and practicing these skills, you’ll train your eyes to see things you’ve probably overlooked in the past regarding color, balance and alignment. More to the point, you’ll be able to take a decent-looking app and understand how to tweak it in order to make it a great-looking one.

GETTING STARTED

Design shouldn’t be an afterthought. As an app developer, you’ve probably discovered that just as an app’s functionality contributes to how it looks, figuring out how the app should look often affects its functionality. Features that seemed important might turn out not to be, once you see that you’re already doing too many things in your navigation menu. Sometimes, after laying out the app screens and writing lots of code for it, you may even realize that you’ve really got two apps on your hands and not just one. Designers have several tricks to help identify these types of issues early on, so you don’t have to change course late in the process and lose valuable time.

WHAT IS THE FOCUS OF YOUR APP?

This may seem incredibly obvious, but you should always start by figuring out what your app does. What do you want your users to be able to accomplish? If you are a visual person, start by imagining a blank canvas and figure out what you want to draw in it. If you are more verbal, you should use a list instead. Either way, just start adding all the features and colors and menus you think you might need.

Next, you want to simplify. This may seem counter-intuitive after all the work you’ve put in, but the single greatest principle of modern app design is that less is always more. Too many features could confuse users. Too many screens are hard to navigate or remember. So simplify.

Draw a line through every feature you don’t absolutely need. Then draw a line through features you think you need, but which don’t fit well with the other things on your list. At the end of this process, you’ll have reduced your app idea down to its purest form.
UNDERSTAND YOUR CUSTOMERS
You need to figure out who your app is for.
Is your app intended for children or adults? Busy professionals or people relaxing at home? Is it for the technically savvy, or are you writing it for someone who needs more help? Understanding these sorts of things will help you picture the overall look, feel, and layout of your app. Busy professionals don’t have time for lots of text, while a less technical audience may need extra clues on how to use your app. Children probably will like bright colors and illustrations more than a millennial will.

Once you know who your target customers are, go talk to them. It may seem like overkill, but the more effort you put into understanding your customers, the better off you will be in the long run. Take video of your intended customers (with their permission, of course), observe them, ask them questions, find out what they would like from your app.

BE INSPIRED—HOARD GOOD IDEAS
Always be on the lookout for inspiration. Collect images, fonts, quotes, textures, and layouts related to your app and your customers—or even just assets that appeal to you at the time—and use them as possible sources of inspiration throughout the process of coding and designing your app. (Of course, don’t actually use them in your app without permission.) Take a look at what other apps in your category are already doing.

Is there anything yours can do better?
Just as important, make sketches throughout the process. You don’t have to be a good artist to doodle (one of the best kept secrets of the digital design world is that many designers are actually poor artists—that’s why they spend their days in front of Photoshop). Get a pen or a crayon and make scratches on paper about your app whenever you get a chance. This is another exercise that will help keep your design mind nimble. Keep all your sketches in a safe place and go through them when you are closer to the end of your app dev cycle. You’ll be happy to discover later that you have a comprehensive history of all the ways your app has changed.
FAIL EARLY, FAIL FAST: PROTOTYPES

Mock up different screen designs based on your customer feedback, either as sketches or wireframes, and imagine the workflow that customers will go through to accomplish their tasks. You can use whatever method works best for you to create these prototypes. If it’s faster for you to do rough interface designs in Visual Studio or Blend, use these tools. But note that you don’t need specialized software for this process, nor do you need to be able to draw. Notecards and sketchpads are just fine.

Even better, try sketching your app design on sticky notes. This way you can easily move screen elements around to see how they look—and, more importantly, quickly put them back if things don’t work out the way you expect.

If you aren’t sure what to try first, take a look at apps that are similar to yours, then see how the workflow might look for your app.

Now try moving screen elements around. If you started with top navigation, you can try left navigation instead. If you have a large header with text, try moving it to the bottom. Buttons and content panels can go just about anywhere. Move them around. Switch out buttons with text; switch out text with links. Experiment. Iterate through this process for every screen until your gut tells you that you’ve gotten your layout to where it needs to be.
VALIDATE YOUR GUT

Now try to find a few test users (at least five or so is usually best) to give feedback on your paper prototype. Just to clarify, this is what’s known as usability testing (and sometimes just market research) rather than any sort of QA testing.

If you can’t find a few people to help, then try to imagine that you are seeing this app for the first time. What’s the initial impact? Do you know, just from looking at it, what the app is supposed to do? Do you know what you are supposed to do next?

If any of the answers you receive are unsatisfactory, hit the sticky notes again and figure out a better way. Remember, you are early on in the process and all you’ve invested so far is a pen, some paper, and a bit of doodling time. Keep at it until you get the results you are looking for.

Thinking about your app’s design at the very start helps you prioritize both the code and the design accordingly. By following the planning guidelines and exercises above, you can catch problems early and avoid expensive mistakes later on.

For more information, check out Plan your Universal Windows Platform (UWP) app.
Understanding typography and UX design

Typography is an ancient discipline concerned with how the formation of letters and words effect usability, readability, and beauty. Making the right typography choices can give your app a feel of accuracy, crispness, and polish. Bad typography choices, on the other hand, are distracting and tend to call attention to themselves.

In the previous section, we introduced some basic design techniques. In this section, we delve into typography and introduce related key terms and principles related. These can enrich anyone’s appreciation of this art and provide that final, professional polish to every app. Chief among these are...

• Kerning
• Character width
• Leading
• Serif and sans-serif
• Font selection (and font families)

KERNING

Kerning (or character spacing) refers to the adjustment of space between two specific letters for legibility. It is one of those little things that you only notice when it is done poorly. In most cases, kerning is handled for you automatically. Should you ever need to fiddle with kerning yourself, as when you’re creating a graphic that includes text, keep these principles in mind.

Kerning is also used to improve the visual balance of a word to make it more aesthetically appealing, especially in banners and headlines. For example, in Figure 2 below, notice how adjusting the kerning between “T” and “y” on the top line of text results in a more balanced product than its counterpart on the bottom.

Good kerning  Bad kerning

Figure 1. Kerning is too tight when two letters are touching one another, which can make it hard to read the word properly. In this example, the r and n are too close, creating the illusion of an m.

Figure 2. Kerning can be used to balance the visual balance of words, which is especially important in banners and headlines.

The key thing to remember when you work with kerning is that kerning is about the perceived rather than the literal spacing and balance between two given letters. In Figure 2 above, for example, the distance between “T” and “y” on the bottom line is literally equal to the spacing between “y” and “p”. It is the spacing on the top line, however, that embodies the ideal appearance of spacing and balance—and this is all that matters. Use your eyes rather than a ruler. If you perceive visual balance in the spacing, so will your customers.
Pay particularly close attention to the kerning in words written all in caps, or where a lowercase is placed next to a capital. Likewise, note that you may need to use rather different kerning to achieve balance between letters of varying shapes—two straight letters (such as “l”) placed side-by-side will usually need a kerning adjustment, unlike two very rounded letters in the same position, such as “g” and “o.”

**Tip:** If you’re having trouble evaluating whether the kerning looks good, try flipping the text upside down! Flipping the text over makes the characters unfamiliar, which can make it easier to evaluate the perceived spacing.

**Figure 3.** Notice how the “A” in Alligator has a tighter spacing than the “l”.

### Character Width

Some fonts, called proportional fonts, have varying character widths. Others, called monospace fonts, have consistent character widths regardless of the character used.

Understanding the role of character width in fonts impacts design in two key ways:
- Font contrast
- Line length

In general, monospace fonts such as Courier New will always stand out from fonts whose character widths vary. The eye will especially gravitate to them if they are used alongside proportional fonts. Keep this in mind when choosing contrasting fonts to use together.

**Figure 3.** A standard typeface such as Caslon has variable character widths compared to the more uniform nature of a monospace typeface of Courier.

Monospace fonts make it possible to calculate the length of a line of text accurately, which is not possible with proportional fonts. Because of this, monospace fonts are handy if you need to know exactly how long a line of text will be.

**Figure 4.** An example of how monospace typefaces produce the same length with the same number of characters even if the words are different.

As with kerning, character width may often be handled for you automatically when you select your font. Still, you want to be mindful of how your selection of a font may impact the character width of your text and thus impact areas like line length.
LEADING

Pronounced with a short “e” sound (LED-ing), leading (or line height) is the spacing between two lines of text. You measure leading from one baseline of text to the next baseline beneath it. A baseline, as you may recall from writing on lined notebooks, is the line your letters sit on. Letters with tails or descenders sit on the baseline but extend their tails beneath it.

SERIF AND SANS SERIF

The difference between serif and sans-serif fonts is core to modern typography. Fortunately, it is fairly simple to distinguish between these two types of fonts: serif fonts have “feet” (represented in red in Figure 7) at the end of their strokes, and sans-serif fonts do not.

This is a sans serif

This is a serif

Figure 7. Unlike serif (bottom), sans-serif (top) lacks adornments at the ends of its character strokes.

For example, Verdana is a popular contemporary sans-serif font, and Times New Roman is such a common serif font that many people will recognize it at once.

Designers often use a combination of sans-serif and serif fonts to help differentiate headlines from the body of text. Sans-serif is frequently used for the headline and serif for the body.

This is by no means a universally held rule, however. Fashion has, from time to time, reversed this wisdom! For example, you can see in Figure 8 below, that MSN News uses serif font for headlines and sans-serif for the body copy.

Welcome to the divergence

I’ve been thinking a lot about convergence recently — the notion that more and more pieces of our lives will somehow be turned into software or consumed by a smartphone in some way. Convergence has been the organizing principle of the tech industry for decades: Microsoft spent the 90s and early 2000s trying to convince everyone into the desktop.

Figure 8. MSN News reverses the current fashion by using serif for the headlines and sans-serif for the body copy.
FONTS (AND FONT FAMILIES)

When selecting font families, try to keep it...

- Limited
- Complementary
- Consistent
- Accessible

In general, limit the number of font families to a minimum (two is plenty, one is often sufficient) and stick to the same ones through the entire app. You could even stick to one font and then use font weights like Light Bold or Italic to provide any needed contrast, or combine members of a single font family like Verdana and Verdana Bold.

If you use too many fonts, the design can seem busy and even interfere with the user’s perception of the visual hierarchy. Take the example of Figure 9 below. This mockup of an app with poor typographic design illustrates how using too many font families can create chaotic results. The menu uses a bold serif font, while its tab counterpart uses a bold sans-serif font. Meanwhile, the serif font used in the story headline also differs from that used in the menu. The sum result is a disconnect between the two UI elements and a cluttered visual hierarchy (the eye gets lost on its journey).

Figure 9. This mockup of an imagined app shows the confusion too many fonts can cause.
If you do use more than one font, ensure the font families complement each other based on their character width, serif or sans-serif nature, and overall appearance. This is something of an art. The fonts should not be so different that the reader wonders why the fonts were chosen (ideally, users should not notice the font at all), but they should still be different enough to add visual interest.

**Georgia**  **Verdana**  **Garamond**  **Impact**

*Figure 10. The combination of Georgia and Verdana (left) share similar values that creates a harmonious pairing. Compare that to the pairing of Garamond and Impact (right) where the heavy weight of Impact vastly overshadows its serif counterpart.*

Finally, when searching for the right font, make sure that it is legible on smaller devices. You probably already know that you should avoid Comic Sans. Also try to avoid fonts that use cursive script, such as Brush Script MT: although they are beautiful, they are difficult to read. You want your user to be able to absorb the content of the text as easily as possible. Of course, there are always exceptions to these rules—use these fonts if you have a compelling reason to do so.

As a practical matter, you are usually safe going with Segoe UI as your sans-serif font in UWP apps. This typeface is very similar to Helvetica, probably the most famous sans-serif font of all time, and is used for the same reasons: crispness, legibility, simplicity. Segoe UI pairs well with the serif font Times New Roman, but also works well with Georgia. Once you are comfortable with these, experiment by switching out your fonts and finding what appeals to you the most. Many professional designers end up with four or five favorite fonts that they simply reuse, two at a time, for many of their projects.

**Tip:** You can find fonts on various websites. The better fonts are not free and come with additional options and support; for example: fonts.com, fonthaus.com. However, you can also find decent free fonts at sites like dafont.com. Whatever type of font you end up using, be sure you have a license to use it on the specific device family for which you need it—and pay attention to the fine print. Likewise, note that using a font that does not appear on the list of recommended fonts for Universal Windows Platform (UWP) apps could trigger a download of font data for your users, which may impact performance or incur mobile data usage costs.

Typography is a discipline that developers do not necessarily need to master in order to use appropriately in their designs. Just knowing the terminology surrounding typography and understanding some of the foundational principles outlined here will make it easier for anyone to apply them.
Using iconography to enhance UX design

Iconography is a visual language used to represent features, functionality, or content. Icons are meant to be simple, visual elements that are recognized and understood immediately. Icons communicate ideas using metaphorical associations: an image of an envelope says “mail,” an image of a house communicates “home,” a diagonally oriented pencil indicates “edit,” and so on.

Figure 11. An example of an icon set.

Figure 12. Sample standard icons.
In many cases, there are already standard icons for the devices and OS your app is targeting. For example, the icon used to represent a calendar and calculator on a Windows 10 device can be seen in Figure 13 below.

Figure 13. Windows 10 device calendar icon (left) and calculator icon (right).

Users of a particular platform are typically already accustomed to and understand these standard icons. So as a general rule, unless you have a specific reason for doing otherwise, you’ll want to defer to these standard icon sets whenever possible to save time and ensure easier user interaction.

WHEN TO USE CUSTOM ICONOGRAPHY

Standard icon sets only go so far, however. In addition to OS or device-specific icons, your app may also need its own set of icons to represent app-specific functionality that the standard sets cannot accommodate without creating ambiguity for the user (e.g. recipe search, movie bookmark, image download).

This goes beyond the user simply not knowing what to do. If the user is likely to hesitate or expend any energy in order to figure out an icon in your app, then you will probably want to use a custom icon.

Tip: Use the text test to determine if your icon is ambiguous. If your icon needs a title or description in order to convey its meaning, consider a different icon or using no icon at all.

If you do need to create and use custom icons for some of your app features, you will want to ensure that all of the custom icons have these two features in common with your other, standard icons:

- Shared thematic style
- Consistent design elements

All of the icons in your app should look as if they were meant to be part of a set or family, even if some of them are customized. Besides being more aesthetically pleasing, consistency in iconography removes ambiguity and helps users distinguish an icon with which they can interact from a graphic or background image with which they cannot (Figure 14).

Likewise, consistency ensures that each icon has equal weight, drawing your users’ eyes equally. Take a look at Figure 15 below. Both the left and right images may look similar, but notice how the icon style on the left has both filled and unfilled shapes. This inconsistency pulls the eye of the user disproportionately to the filled icons—video, mail, and photo. In contrast, check out the icon set on the right. Those icons are consistent in both their thematic style and design elements. Each one engages the eye equally.

Figure 15. Although these two images may look similar, the inconsistent use of fill on the left draws the eye to the filled icons. A consistent visual language, however, such as that on the right, allows the user to become familiar with the visual cues you want them to find.
WHERE TO FIND ICONS

Like fonts, icons are available for download. Icon sets are designed by professional artists and made available for you to use in your app. You may choose icon collections that are free (like vectoricons.org and freeiconsdownload.com) or icons that can be purchased (such as those on popular sites like istockphoto.com, shutterstock.com, and iconshock.com).

Either way, be sure that your use of them falls within the license guidelines and that you properly attribute them according to the requirements provided on the download site.

Iconography may be the aspect of app design that comes closest to actual poetry because it depends on visual metaphors only to communicate meaning. Some associations are direct, such as using a stylized picture of a camera to represent a taking a photo with a webcam. Some are less so, such as using a shopping bag to represent the Windows Store in Windows 10. What makes iconography fascinating is that once users get used to these associations, they become automatic and effortless in a way that glancing over groups of letters can never be.
Adding color to your design

What is your favorite color? Most of us have been asked this question for as long as we can remember. And for most of us, this decision probably marked our first and foundational experience with having an artistic preference.

Color matters. We have already touched briefly on other topics that have an artistic as well as a scientific side, such as typography and iconography. But whereas the effects of typography and iconography are subtle, the colors in your app have an immediate, emotional impact on your users.

In this section, we will deal with this emotional element of color in UX design, general rules of thumb for combining color, accessibility issues to consider, and finally, tips for finding the right colors to suit your specific app.

COLOR AND EMOTION

Colors have an instantaneous effect on the brain and human emotion. Green rooms tend to make people feel calmer while red has been shown to enhance physical reactions. Although the influence of specific colors tends to vary depending on context as well as cultural background, the anecdotal evidence that certain colors can be used to influence people has been convincing enough that advertisers and product companies take color very seriously.

Here are some color attributes you should be familiar with:

- **Red** is a vibrant and activating color associated with passion and danger.
- **Orange** evokes nature as well as energy.
- **Yellow** projects joy, intelligence, and positivity.
- **Green** suggests renewal and is often found in financial apps.
- **Brown** is natural and organic and can often serve as a good contrast for more vibrant colors.
- **Blue** is often used to project calmness, security, and professionalism.
- **Violet** evokes royalty and luxury.

These attributes should be taken as guidelines only. The meaning of a color can shift dramatically depending on the other colors combined with it and the context and imagery it’s associated with in your app. If you are interested in a deeper dive into the relationship between color and emotions, you may want to explore the Color In Motion website. [www.mariaclaudiacortes.com](http://www.mariaclaudiacortes.com)
COLOR THEORY

Isaac Newton came up with modern color theory and even created the modern color wheel. You probably discovered color theory on your own, though, using finger paints and crayons. There are three primary colors used when designing for electronic devices, red, green, and blue. By combining these in an additive system, we can get three secondary colors:

- red + blue = violet
- green + blue = cyan
- red + green = yellow

Colors can be placed on a wheel, as Newton did, with the secondary colors placed between the primary colors. This is a natural way to position colors and highlights some interesting relationships (fig. 18):

- Complementary colors (a). Colors that are directly opposite each other on the color wheel.
- Analogous colors (b). Colors on either side of a given color on the wheel.
- Split complementary colors (c). Colors that are the analogous colors of a complementary color.
- Triad colors (d). A set of three colors that are equally spaced apart on the wheel, each 120 degrees from the others.

Colors are also divided into warm and cool tones. The distinction between warm and cool colors is related to the anecdotal effects of color on the emotions discussed above. The warm colors extend from yellows at one end to reds at the other. The cool tones are blues, greens, and violets. Colors such as black, white, and the grays are neither warm nor cool. Instead, they are known as neutral colors. The remaining shades—such as browns, tans, and pastels—are called near neutrals.

UWP color codes are based around the red, green, and blue (RGB) model of color creation—and so are computer screens, for that matter. Any color on the wheel can be derived from a value from 0-255 for each primary color component for a total of 16,777,216 colors.

Tip: Xbox knocks off the ends of the spectrum, supporting somewhat fewer color choices. HoloLens can’t use pitch black (0, 0, 0) because this represents transparency in Windows Holographic.
Because you will occasionally encounter it, you should also become familiar with the HSL system, which stands for hue, saturation, and lightness. It offers a different mathematical model for representing colors. HSL (as well as a variant called HSV) is popular with graphics programmers and is probably already known to you if you’ve ever used a color picker. A hue can be thought of as a pure color. Saturation is how intense the hue is. A lightness of 100% applied to a hue gives you white, whereas a 50% lightness gives you the pure hue.

COLOR CONTRAST AND ACCESSIBILITY

Contrast describes how far apart two colors appear to be. Color contrast is important for an app that contains text because a high contrast will make your app easier to read. It is also important for your visual hierarchy, since high contrast will more clearly differentiate different areas of your layout.

So far, contrast seems like just an aesthetic problem. Contrast becomes an accessibility problem, however, when your app is used by people with poor vision or by people who are color blind. Color vision deficiency affects 8 percent of men and 0.5 percent of women worldwide. Because of this, color contrast isn’t something you can ignore.

There are two ways to evaluate the color contrast in your app. One way is to desaturate your colors until they are effectively grayscale. If your text is still readable and your visual groupings are still clear, then your color contrast is effective. Alternatively, you can use a color contrast analysis tool, like this one on the one above found on www.colorsonteweb.com. Use this site to plug in your color values and receive an immediate evaluation of your Universal Windows Platform (UWP) app’s visibility.
COLOR SCHEMES

There’s a lot of background that goes into understanding color and using colors effectively. Fortunately, there’s a shortcut to all of this color stuff that you can start using right away. Adobe provides a free tool at [https://color.adobe.com](https://color.adobe.com) that will allow you to quickly pick a color scheme that you like.

We each have an intuitive understanding of how colors affect us emotionally. The goal of this section has been to provide you with the confidence to trust your own instincts when it comes to hues. With the color concepts provided, you as a developer are in a position to better understand why you prefer one set of colors over another and are well-equipped to have informed discussions about color schemes based on color psychology, color theory and accessibility.

This app will let you select any of the color relationships discussed in the color theory section above to select your color scheme: triad, complementary, analogous, or split complementary. It will then give you the color values for this color scheme at the bottom as both RGB and HEX values, which you can then plug into your UWP app. All you need to do is start with a dominant theme color that will create the right emotional context for your app using the descriptions in the color and emotions section. You can also play around with the tool until you find a main color that appeals to you and reflects the purpose of your app.

![Figure 22. Screenshot of Adobe Color CC tool.](image)
Accessibility and your app design

Accessibility is about making your app usable to the largest possible audience. For some apps, accessibility is required by law. For others, it’s part of the service you are offering to a specific audience and a way to make your app more generally appealing.

Choosing to incorporate accessibility features is a good idea no matter what your motivation. Thinking about accessibility, in turn, will help you to become a better designer because you will be considering the user experience much more broadly for a greater variety of users.

**BE ACCESSIBLE**

Accessibility options include features relating to mobility, vision, color perception, hearing, speech, cognition and literacy. However, you can address most requirements by providing:

- Support for keyboard interactions and screen readers
- Support for user customization, such as font, zoom setting (magnification), color, and high-contrast settings
- Alternatives or supplements for parts of your UI, such as audio descriptions of text for those who are visually impaired

Standard Windows controls already have Microsoft UI Automation support and are accessible by default. They require fewer accessibility attributes that are app-specific. If you want to create a custom control, you can add similar support by using custom automation peers.

In the UI design, here are some steps you should take to ensure your app works well with the following scenarios:
<table>
<thead>
<tr>
<th>Scenario</th>
<th>STEPS TO TAKE</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCREEN READING</td>
<td>Users of this feature rely on a screen reader like MS Narrator to help them create a mental model of your UI. To help them interact with your app, you need to provide information about its UI elements, such as name, role, description, state and value. Learn more about exposing basic accessibility information.</td>
</tr>
<tr>
<td>KEYBOARD ACCESSIBILITY</td>
<td>Allow users to interact with all UI elements by keyboard only. This enables them to:</td>
</tr>
<tr>
<td></td>
<td>- Navigate the app by using tab and arrow keys</td>
</tr>
<tr>
<td></td>
<td>- Activate UI elements by using the spacebar and enter keys</td>
</tr>
<tr>
<td></td>
<td>- Access commands and controls by using keyboard shortcuts. Learn more about implementing keyboard accessibility.</td>
</tr>
<tr>
<td>ACCESSIBLE VISUAL EXPERIENCE</td>
<td>Some visually impaired users prefer text displayed with a high contrast ratio. They likely also need a UI that looks good in high-contrast mode and scales properly after changing settings in the Ease of Access control panel. Where color is used to convey information, users with color blindness need color alternatives like text, shapes and icons. Learn more about supporting high-contrast themes and about meeting accessible text requirements.</td>
</tr>
<tr>
<td></td>
<td>Important: To prevent seizures, avoid elements that flash. If you must include flashing elements, do not let them flash more than three times per second.</td>
</tr>
</tbody>
</table>
There are other design-related ideas you can employ to make your app more accessible. For example, try to reduce UI gestures to make common tasks require less button pressing.

Provide multiple ways to interact with the same control. For example, activate UI elements by use of the keyboard, rather than just by touch or click. A control that only supports mouse interactions would be extremely difficult, if not impossible, for a visually impaired person to use. To make a control accessible to the visually impaired, you must provide keyboard shortcuts. Also, let users navigate your app using the tab and arrow keys.

Consider creating a tab flow diagram to help you plan how the user would progress through common tasks using only the keyboard. The tab order is also important because it indicates what order screen readers should present the text to the user. For more information, refer to the eBook “Engineering Software for Accessibility”.

Tip: Include only interactive elements in the tab order.
NOT EVERYONE HAS PERFECT VISION

Design your text and UI to support a high-contrast theme. While color is important, it must not be the only channel of communicating information. For example, users who are color blind would not be able to distinguish some color status indicators from their surroundings. Include other visual cues, preferably text, to ensure that information is accessible.

Figure 23. The screen on the right represents, for those viewing this blog without colorblindness, what those who are colorblind would most likely see – reds and greens are indistinguishable.
Imagine you had to work with a black and white printout of your app's screen. The values of the colors—how dark or light they are on a grayscale—help you tell them apart. This is important to keep in mind when you are selecting colors for highlights and calls-to-action, which you read about in the section on visual communication and visual cues. You can also add textures to help distinguish the colors—but try to keep them subtle.

Figure 24. By adding text indicators, users who are colorblind are able to recognize the various color options even though they are not able to see it.
Optimize your UWP app design for touch input and get basic support for mouse, pen and touchpad by default. Visit MSDN to learn more about designing interactions that are easy to use.

**SCALE**

Allowing users to zoom and resize elements can be helpful to people with visual impairment, especially for images that include words. Ensure that text and UI scale appropriately when Ease of Access settings are changed. However, take care not to start with a font size that is too small in general for many users. Everyone’s vision deteriorates as they age; your app should be available to users of any age.

To allow for differences in vision, provide scaling options for your users in your app settings. They might change the font size for easier reading or want the ability to shrink or enlarge the UI.

**Tip:** Use vector images (SVG) rather than raster images as they can become pixelated when enlarged, distorted when shrunk. Vector images look proportional and clear at any scale.
Once you believe you have your design working, be sure to test it on your target device(s). The Windows Software Development Kit (SDK) includes accessibility testing tools such as AccScope, Inspect and UI Accessibility Checker. Testing the design should help you identify any areas that need correcting, as well as any opportunities presented on different screen sizes. For more information about these tools, see the article about accessibility testing on MSDN.

If you want your app to look good on very large screens, you may want to include optional larger images, add more whitespace, add rows or columns, or incorporate more navigation options without using submenus. You could also take advantage of the extra space by adding something like an overlay on part of the screen to provide more information, such as details about a selected item or a view of the user’s cart.

Figure 25. When zoomed onto the section of the illustration, vector images produce clear sharp edges compared to the pixelated and blurriness of the raster image.
Accessibility design is really just usability design for a larger audience. While the Universal Windows Platform (UWP) can take care of some of this for you, you will find that a bit of thoughtfulness in incorporating built-in controls will actually go a long way toward making your app more accessible and usable for everyone.
Communicating with your users

While a Universal Windows Platform (UWP) app should be visually appealing, it should also project the voice of the original app creators from time to time. After all, a good user experience is ultimately a human-centric experience.

Communicating effectively with your users requires the following three virtues:

• Be friendly (and compassionate)
• Provide guidance
• Project a gentle tone

Friendliness, helpfulness and a gentle tone should all combine to provide a smoother, more seamless experience for your users even in the toughest situations.

FRIENDLINESS

An important aspect of compassionate design is the character of the communication between you and your user. You want users to feel as if there are real people behind the app who care that their experience is a good one.

First, the app should demonstrate a considerate attitude toward the user’s success—both in terms of messaging for errors, user actions and potential experience improvements. Second, there should be accommodations for the users themselves when they do the wrong thing (possibly an indicator that the design was not sufficiently intuitive).

Figure 27. Message notifying the user that they have logged in.
Messaging, especially in regards to errors, should be clear and generally upbeat. In Figure 28 below, for example, notice how MeeDJ uses clear messaging to convey that the playlist is loading—a positive affirmation of the desired behavior. Without this messaging, the user might assume that the black screen and delay are a failure of the app to function rather than a momentary and fully expected delay. Here (fig.28), just a little bit of text helps to avoid a potentially frustrated user who might otherwise abandon the app altogether.

**Note:** Messaging requires balance: while it is important to reassure your users, you also want to be careful not to disrupt the user experience unnecessarily.

*Figure 28.* MeeDJ uses clear messaging to indicate why there is a brief delay.
A little forethought can also go a long way toward reducing the frequency of issues and ensuring the user is treated gently if and when they do happen. For example, you can anticipate that users will occasionally tap on the wrong spot. You can prepare for this by making the touch targets large enough to accommodate some imprecision. And, if users do still manage to tap on the wrong spots, you can ensure they have a means for easy recovery, as Fused does in Figure 30 below. Here, the app (1) prompts the user to confirm the action before proceeding, and (2) reminds the user that proceeding will result in a loss of changes.

A word here about the attitude of design forgiveness: Never blame the user for errors. What we mean is that if the user did something wrong, it is possible that the interface wasn’t clear or accommodating enough. If the user gets confused it is possible that the user experience might need to be friendlier. Offer your users no reason to seek out an alternative to your app—instead, take each opportunity to make your users feel good about using yours.

When it comes to the messaging of errors, remember numerous informative, but ultimately un-actionable messages from the app are not necessary (Figure 29, left). The focus should be more on getting the user back on track than on fully documenting what’s happening for them in painful detail (Figure 29, right). The user doesn’t really care what happened. They just want the app to start working again.

Figure 29. The image on the left contains too much information. The focus of an error message should be letting the user know that something occurred in the app and action needs to be taken in order to get the app back on track (right).

Figure 30. Fused knows that users occasionally select the wrong actions, particularly when they are new to an app and offers them a way out.
PROVIDE A HELPING HAND

Speaking of which, remember that no one likes to feel stupid or confused. One way to create your users at crucial moments or with critical features is to take them by the hand and show them around.

A powerful example of this principle in action is the initial walkthrough, which lets users know that you have thought carefully about their experience and want to make it as easy for them as possible. Keep in mind, though, the walkthroughs should be in addition to an intuitively designed app and should not be used as a crutch for poorly planned design.

Microsoft’s OneNote does a good job of this by providing simple instructions covering how to use the app. In Figure 31, squiggly lines, arrows and exclamation points provide a sense of fun and informality to the walkthrough. Even though OneNote is a very sophisticated app, the walkthrough portrays it as something different—an approachable, easy-to-use app you can use as much or as little of as you want.

OneNote also creates an ongoing sense of discovery. Different aspects of the app can be explored over time. This creates the powerful suggestion that OneNote is more like a game than a line-of-business app. You learn how to use it by playing with it rather than by reading a manual.

Add pages inside of each section:

_add a new page in the section Trips_

Add pages!

(Special is over there)

Figure 31. OneNote uses informal illustrations to draw attention to the user interface.

For more tips, check out 30 second videos

Clip from the web

Plan a trip with others

Search notes instantly

Write notes on slides

Figure 32. Learning OneNote is an act of exploration. You don’t have to learn it all at once.

Consider the role of guidance in your app. It might be a full walkthrough upon first launch or access of critical features, helpful pop-up tips or a little question mark on each page for easy access to guides. Whatever you choose, remember that the primary purpose is to reduce user confusion and show that you, as the app creator, care about the user experience and want them to succeed.

Tip: You should always provide your users with a way to dismiss the guidance if they feel they don’t need it. By the same token, there should also be a checkbox in your app settings to bring the guidance back if a user changes his or her mind.
The important thing in setting tone is to be consistent and sincere. You can’t switch back and forth between being professional and being playful, which might create confusion with the users, as well as a sense that you are talking down to them. Instead, think of your tone as a way to convey the personality of the people behind your app.

Professional:
Loading playlist...

Playful:
Hang on. We’re just loading the playlist for you...

Most of all, make sure the tone feels comfortable to you. If you ever feel as an app creator that your app’s tone doesn’t reflect who you are, your user is going to feel it, too. The best way to communicate with your user on a human level is to be yourself.

Good app design is about human-to-human communication. While words should be used sparingly in an app, you want them to have maximum impact when you do use them—so choose your words carefully! Choose words that convey your desire to help your user learn your app. Choose words that soften the impact when accidents happen, either through user action or technical glitches. Finally, choose words that communicate the tone of your app and the personality of its creator(s).
Visual communication and visual cues

In this section, we introduce you to the science of communication through visual cues. We’ll look at:

- Visual cues in general—typical examples of visual communication
- Affordances—visual cues that tell us how to interact with an app
- Calls to action—visual cues to complete transactions

Visual cues are visual clues you leave for your users so they can spend less time figuring out what they need to do and more time simply getting things done.

VISUAL CUES IN GENERAL

If your app’s users hesitate because they don’t know what to do next, or don’t understand how to use your navigation, or can’t figure out if a bit of text is intended to be content or a button, they may close the app and never come back.

A great visual cue unobtrusively helps users understand whether they are in the right place for what they would like to accomplish. For example, as you can see in Figure 35, the text and icons are cues for Flipboard’s users, but note that they do not distract from what Flipboard is emphasizing—the image and its caption.

Figure 35. Flipboard’s hero image utilizes text and intuitive icons as cues for the user. At the same time, the cues don’t detract from the two main points of focus: the photo and the caption.
Visual cues can be executed in a variety of ways:

- Providing text instructions
- Using size, color and contrast to draw the eye
- Placement in a prominent location on the screen
- Using lines, arrows and unambiguous icons

For example, the visual cues highlighted in Figure 36 below demonstrate several of these characteristics in action.

Visual cues help your user to quickly understand what is happening in your app and to see what is important and what is not. They also orient users within the app and show them the things they can do in it.

Figure 36. This mock-up of an actual app shows several key principles of visual cues in action.

Icons are subtly added to act as call to actions. Large hero placed on the left side indicate prominence. Secondary content has a distinctive visual treatment to stand out.
AFFORDANCES

An affordance is a special kind of visual cue that tells the user how to interact with objects on the screen. Visual cues in the real world create relationships between people and things. For instance, a doorknob affords twisting, a cord affords pulling, a button affords pushing.

Just as real world affordances don’t require any thinking, in-app visual cues similarly should clearly communicate to users how they interact with things. Clear digital affordances let the user know they should tap, drag, drop, pan, scroll, or pinch.

In the app world, perceived affordances generally rely on conventions to convey meaning. Buttons change their appearance when pressed. Draggable objects have handles (Figure 37). Drop zones change color when you drag items over them. Scrollable areas have arrows (Figure 38).

Don Norman, the great advocate of user-centered design, states that there are two things that matter in designing an easy-to-use affordance:

- Whether the desired control can be readily perceived and interpreted
- Whether the desired action can be discovered

As long as you rely on conventions like standard button templates, you’ll be fine. If you’re introducing a new sort of interaction and can’t rely on conventions, you can still base your affordance on real-world objects. After all, this is where our conventions originally derive from.

Figure 37. Handles indicate that the object has a draggable action.

Figure 38. Arrows (in purple circles) indicate that the calendar area is scrollable.
CALLS TO ACTION

A call to action (sometimes known as a CTA) provides another example of how to use visual cues. Calls to action, such as sign-in/sign-up buttons, are transactional in nature. You’ll see them frequently in free-to-play apps that want you to perform an additional task, like clicking on an ad or clicking on a purchase button.

CTAs usually have wording and design features that invite users to see the button and act immediately. While you may sometimes need to provide detailed instructions to users for calls to action, typically an assertive command like “Become a member” suffices.

**A call to action...**

*invites the user to give information, such as an email address or profile photo*

*urges the user to do something, such as “buy” or “download” or “save”*

*offers the user an opportunity, such as a chance “to learn more”*

Calls to action use one or more of the following characteristics to visually clue the user into noticing them and understanding their purpose:

- Contrasting color
- Noticeable difference in scale
- Larger font
- Noticeable margins

In Figure 39 below, the call to action button in the drop down navigation of LinkedIn’s Lynda app is bright blue. Its color and shape distinguish it from other clickable or tappable elements. This same treatment repeats throughout the app to establish a pattern and make it easier to identify calls to action. This trains users to understand what these buttons are for.

![Figure 39. The call to action for LinkedIn’s Lynda app is bright blue with a shape that differs from those around it, drawing attention to itself. This visual clue alerts the user to the importance of the button.](image-url)
You can see a different take on the call to action in Figure 40. Additional content is available by selecting “Read More” below the introductory paragraph. To identify this as a call to action, it is in blue and bold style to set it apart from the content itself.

Author
Brenda Romero
Brenda Romero is an award-winning game designer, artist, writer, and creative director who entered the video game industry in 1981 at the age of 15. She is the longest continuously serving woman in the video game industry. Brenda has worked with a variety of digital game companies as a game designer or creative director, including Atari, Sir-Tech...

Read more

While visual clues should be noticeable to the reader, and calls to action should be doubly so, it is also important to ensure that visual clues and calls to action are not jarring. Jarring visual clues (Figure 41) and calls to action can actually frustrate the eye of the user and distract them from completing their goal.

Great visual communication is at the heart of great visual design. It goes beyond simply making your app attractive and gets to the basics of making your app usable and effective. In this section we reviewed three topics in visual communication: visual cues, affordances, and calls to action. By keeping these in mind as you design, you’ll make your apps more effortless to navigate, which in turn, helps your users get more done.

Figure 40. The differing style used for the “Read more” button calls attention to itself and invites users to take action accordingly.

Figure 41. Although the tool selection menu matches the playfulness of this kid’s app, the design choices here make using the app far from effortless. The overall design is so jarring that the user’s eye gets lost in the noise. As a consequence, the menu blends in with the image.
Designing for intuitive navigation

In this section, we will look at a topic that has much less to do with solving UI design problems and much more to do with solving user experience (UX) design problems. We’re going to look at navigation.

Navigation involves structuring your app in a way that promotes ease and efficiency, not merely design aesthetics. Unlike the other topics covered so far, which rely in part on personal taste as well as an awareness of changing fashions, navigation—and UX in general—navigation tends to rely heavily on usability research.

We will explore the three key principles that both research and experience have shown underpin good navigation design:

• Consistency—meet user expectations
• Simplicity—don’t do more than you need to
• Clean interaction—keep out of your users’ way

Keeping these principles in mind as you design will help you to achieve the ideal navigational structure—one that the user never notices.

CONSISTENCY

Navigation should be consistent with user expectations, leaning on standard conventions for icons, location and styling.

For example, in Figure 42 below, you can see the spots where users will typically expect to find functionality, like the menu and back buttons. The user expectation that the menu button will be at the top-left of the screen is so strong that you can even consider using a non-traditional icon to represent it, although the traditional “hamburger” icon is often the preferred choice across most platforms. For the back button, it is better to stick with Windows convention and keep it in either the leftmost spot or, if there is a Windows button, in the second-to-the-left spot.
Placement of navigational elements should also change for different device families. For example, on tablets and laptops/PCs, the navigation pane is usually placed on the left side, whereas on mobile, it is on the top.

Figure 42. Users expect to find certain buttons in certain places—e.g. the menu in the top left, and for UWP apps, the back button in the leftmost or second-to-the-leftmost spot. Sticking to these standard conventions helps users interpret the meaning of the buttons.

Figure 43. Different device families have their own conventions for navigational elements. For example, the navigation pane typically appears on the left side of the screen for tablets, but up top for mobile devices.
SIMPLICITY
Another important factor in navigation design is the Hick-Hyman Law, often cited in relation to navigational options. This law encourages us to add fewer options to the menu (see Figure 44). The more options there are, the slower user interactions with them will be, particularly when users are exploring a new app. The same law can be applied to media content. Rather than overwhelming the user with a vast selection of media options, consider providing brief tidbits for users to explore if they choose.

CLEAN INTERACTION
The final key characteristic of navigation is clean interaction, which refers to the physical way that users interact with navigation across a variety of contexts.

This is one area where putting yourself in the users position will inform your design. Try to understand your user and their behaviors. If you’re designing a cooking app, and you’re expecting it to be used in the kitchen, you might want to take into account that the user will probably want to avoid using food-covered fingertips to navigate to the next cooking step. Instead, the user might use a knuckle, the back of his or her hand, or even an elbow. This should influence the size of your touch targets and the appropriate spacing between navigational elements, at the very least.

You should also keep in mind which areas of the screen are considered easy to reach. These are known as interaction areas. In the mobile device illustration below (Figure 45), for example, the blue area represents the optimal touch points for users (in this case, a user with the phone held in her left hand). Here, users expend the least amount of effort to interact—remember that most users hold their phones with their left hands and interact with their thumbs. Correspondingly, the dark grey region requires somewhat greater effort for interaction than the blue, and the light gray area requires the greatest amount of effort overall.

Tablet devices introduce an additional complexity because users have multiple ways of holding their device. Typically, users grip a tablet with both hands along the sides. Figure 46 below shows the interaction area for the most common pose and grip of a tablet. Keep in mind as you design your navigation, however, that tablet users often switch between posing their devices in landscape and portrait orientations. Finally, note the alternative ways you yourself interact with tablets and consider whether your navigation is convenient for those scenarios, as well.

Figure 44. On the left, notice there are fewer options for the user to select, whereas on the right, there are several. The Hick-Hyman Law indicates that the menu on the left will be easier for users to understand and utilize.

Figure 45. Mobile device interaction area.

Figure 46. Tablet device interaction area.
RULES OF THUMB

Several rules of thumb help designers to encapsulate consistency, simplicity and clean interaction in their navigation design. Most of these come from the web design world and will apply to touch and non-touch devices. As with any rule of thumb, use them as starting points and tweak as needed.

1. Avoid deep navigational hierarchies. How many levels of navigation are best for your users? A top-level navigation and one level beneath it is usually plenty. If you go beyond three levels of navigation, then you break the principle of simplicity. Even worse, you risk stranding your user in a deep hierarchy that they will have difficulty leaving.

2. Avoid too many navigational options. Three to six navigation elements per level are most common. If your navigation needs more than this, especially at the top level of your hierarchy, then you might consider splitting your app into multiple apps, since you may be trying to do too much in one place. Too many navigation elements in an app usually lead to inconsistent and unrelated objectives.

3. Avoid pogo-sticking. Pogo-sticking occurs when there is related content, but navigating to it requires the user to go up a level and then down again. Pogo-sticking violates the principle of clean interaction by requiring unnecessary clicks or interactions to achieve an obvious goal—in this case, looking at related content in a series. (The exception to this rule is in search and browse, where pogo-sticking may be the only way to provide the diversity and depth required.)

Figure 47. Pogo-sticking to navigate through an app—the user has to go back (green back arrow) to the main page in order to navigate to the “Projects” tab.
The goal of navigation design is to help your user move through your app without noticing your navigation structure at all. This is accomplished by making your navigation design simple and clean, typically through the reuse of the navigation idioms that everyone else uses whenever you can. By making your navigation uninteresting and consistent with standard conventions, you are actually helping your users to navigate your app intuitively.

4. Mind your thumbs. Your users will typically use their thumbs to navigate your app. At the same time, they are also using their thumbs to grip their device, leading to a bit of clumsiness in interaction. To throw an extra wrench into this, thumbs are relatively big compared to other fingers and tend to hide the visual element with which a user is trying to interact. The technical term for this occurrence is occlusion. Keep occlusion in mind as you create your navigation structure.

The goal of navigation design is to help your user move through your app without noticing your navigation structure at all. This is accomplished by making your navigation design simple and clean, typically through the reuse of the navigation idioms that everyone else uses whenever you can. By making your navigation uninteresting and consistent with standard conventions, you are actually helping your users to navigate your app intuitively.

Having an icon (swipe gesture in green) helps to resolve this issue, as you can see in Figure 48.

Figure 48. You can resolve some pogo-sticking issues with an icon (note the swipe gesture in green).
Polishing Your App with Animations and Audio Cues

There are plenty of well-designed apps out there. Often what distinguishes a great Universal Windows Platform (UWP) app from a good one is the small touches, like animations and audio cues. In this section we will provide guidance on some of the subtle nuances that can help you take your app to the next level. It will also extend some of the topics we have already covered in this series, such as color, navigation and visual communication, and visual clues.

As you apply animations and audio cues in your app, it is important to remember that these elements are no longer optional. App users expect regular feedback that lets them know their app is working and in constant motion. An app that lacks visual and audio flare is often seen as unresponsive or glitchy.

**ANIMATIONS THAT CALL ATTENTION**

Animations are a dynamic way to add flare to your app, and they can be applied in many different ways. Sometimes, the purpose of an animation is to draw your attention, as with a call to action. At other times, animations are used as transitions between different states or screens as you navigate through your app. As a general rule, you want the animations to be subtle, smooth and unobtrusive.
<table>
<thead>
<tr>
<th>ANIMATION</th>
<th>POLISHED</th>
<th>OKAY</th>
<th>OVERBLOWN</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIMPLE BUTTON CLICK</td>
<td>Shift in color or slight variation in light/shadow</td>
<td>Skeuomorphic button up/button down</td>
<td>Special effects involving particles or light effects</td>
</tr>
<tr>
<td>CARD CLICK &amp; RELEASE</td>
<td>Card rises from the surface, and shows slight motion toward the selected action, then flips around to reveal the result</td>
<td>Card enlarges to show content</td>
<td>Card performs an effect that calls attention to itself, rather than to the content</td>
</tr>
</tbody>
</table>

Animated assets possess their own style in the same way that design assets do. Just as you want your icons and buttons to look like they come from the same set or template, you want your animations to look as if they share the same style. Additionally, the animations should share a similar pattern of motion when they are applied to elements or events.

Flipboard takes advantage of in-app animations with every page turn. As users move from page to page, the pages flip over as if they were in an actual magazine. Animations do not have to be limited to reactions to user actions, either. They can be used for notifications, confirmations and indicators. For example:

- App start or close
- App action beginning or finishing
- When a notification is received, such as a message from your app or another app

Animations do not need to be elaborate, time-consuming, or expensive to design. In fact, simpler is usually better. You can do much with simple transformations: size, color, opacity and position. For example, a button state change can be communicated with a small offset in position and a slight fade effect (a change in opacity).

![Flipboard is Your Personal Magazine](image)

*Figure 49. Flipboard uses animations to suggest and confirm user interactions*
HOW LONG SHOULD AN ANIMATION STICK AROUND?

Like a good joke, animations are all about timing and pace. Like a good guest, animations should leave before they become an annoyance. A movement that lasts too long will make your users impatient. A motion that is too quick can agitate your users. The goal of a finely-tuned animation is to find that happy medium between too quick and too slow.

Fortunately, motion designers have actually put a number to this happy medium. A good animation or audio cue will last between 150 and 350 milliseconds—in other words, around a sixth to a third of a second. Once you find an interval that works for you, try to use it consistently throughout your app.

ANIMATIONS THAT TRANSITION

App navigation is enhanced through the use of transition effects. Having one page disappear while another takes its place is simply too jarring. Instead, one page should either fade or slide out smoothly while the other fades or slides in.

You already know that you want animations to last from 150 ms to 350 ms. Since transitions actually combine two motions, one leaving and one entering, a full transition will last from 300 ms to 700 ms. To make your transitions even more pleasing, however, you should vary the rate at which the transitions occur by adding an easing effect.

Easing reflects how movement occurs in the real world. Objects do not typically go from inactivity to a constant velocity and then back to a state of inactivity. Instead, they ease into motion, going slowly at first and then accelerating to achieve a desired speed. Adding easing to your transitions makes animations appear more natural and organic.

There are many mathematical formulas governing easing effects. As with many things, however, the simplest approach is often the best. If you can keep track of the difference between an Ease-in and an Ease-out, you are most of the way to mastering transition animations.

Figure 50. Ease-in your animations.

An ease-in effect starts off slow and then speeds up. It basically eases into the motion.

Figure 51. Ease-out your animations.

An ease-out is the diametrical opposite of an ease-in. It basically starts off fast and then slows down. When in doubt, the ease-out should be your go-to when reaching for an easing. The ease-inease-out, as you would expect, is slow-fast-slow. It imitates the motion of a vehicle such as a train or a car, and can be a very satisfying effect.

Figure 52. The always useful ease-in, ease-out.
There are also variations of each of these easing effects such as sine, cubic, quad, and quint. You should play with these different effects and find something that you like. Once you do, try to use your preferred easing effect consistently throughout your app.

**SOUND EFFECTS**

Every place you have an animation you could also employ audio, either instead of or in addition to the visual effect. Again, you want to go for subtlety rather than fanfare. If your audio effect seems fun or impressive, imagine the effect it might have if repeatedly played while a frustrated user tries to successfully complete a task. For example, if your error sound is an “Uh oh,” you might just drive your user away. Conversely, some people will find any audio distracting, so it is a best practice to ensure your app has a setting that allows users to disable all audio cues and background audio in your app.

**BEST PRACTICES FOR SOUND DESIGN**

As user patterns emerge in app design, sound effects have become simple, impactful, and purposeful. The touch of a keyboard, turning a device on/off, notification of an email/text message, the refresh of an application. When it comes to audio UI design, it is always wise to ask yourself, what does this sound bring to the app?

In game design, for example, sound effects and background music help to set a mood and make the experience richer. In other types of apps, however, numerous sound effects can distract the user from accomplishing their tasks. Use sound effects with discretion for apps, suit them to the nature of your app, and ensure they feel intentional.

In some cases, you might also want to consider adding subtle haptic feedback. This is an obvious option for apps that are designed for a quiet setting, but can also be valuable in providing a tactile experience where a user may otherwise miss something. For example, a fleeting pulse to confirm a button press, the success of a drag and drop gesture, or even the appearance of something important in a reading list. Different tactile patterns could be used to convey different feedback.
IN UWP APPS, AUDIO OFTEN GOES UNDERUSED

If you want your UWP app to stand out, use audio. For inexplicable reasons, audio is simply not included in many UWP apps even though it is an extremely useful tool for providing affordances and feedback, especially in touch apps. While your best efforts at creating visually appealing animations may get overlooked due to someone’s thumb occluding the user interface, it happens to be a fact that audio feedback can’t be occluded. If you need to indicate to a user that their button press was unsuccessful for some reason, or you simply want to provide your user with encouragement following a successful tap, subtle audio cues will do the trick.

Audio cues also provide an additional layer of depth to your app that most other Windows Store apps simply lack. In a competitive app market where it can often be hard to stand out from the crowd, introducing a UI feature that your competitors lack could make that all important difference.
UNDERSTANDING WHAT KIND OF INSPIRATION YOU’RE LOOKING FOR

Are you creating a line-of-business app, a game, or a utility? This has a lot to do with deciding what type of inspiration to seek. Users interacting with a line-of-business app may not be thrilled with a quirky, playful design. Casual gamers, on the other hand, might find quirky and playful to be in perfect alignment with their interests. Knowing the kind of inspiration you need will help you narrow your targets as you hunt down interesting sources for new, creative ideas.

Design thinking: finding your inspiration

One of the biggest differences between developers and designers is that development is something developers do while design is something creatives are. To think like a designer means constantly honing one’s tastes and looking for new ways to be creative.

In earlier sections, we covered concrete design topics like typography, iconography and visual cues. Now we’re going to switch things up a bit. We’re going to look at design thinking, inspiration, and creativity—and the right-brain techniques these involve when it comes to designing the user experience of your Universal Windows Platform (UWP) app.

The best way to get started is to begin the same way creatives and designers do: by asking what, where, and how.

1. What are you looking for?
2. Where are you going to look for it?
3. How are you going to store it once you’ve found it?

What, where, and how are the secrets to finding what you need for inspiration and then arranging those inspirational resources in such a way that you can access them easily. We’ll explore them one by one.
Think about breaking out your search by the different components of your app. You can look for inspiration for your navigation separately from your color theme or your grid system. Rarely will you find something that hits all your hot buttons in every category.

Don’t be afraid to mix and match design inspirations from different sources. Sometimes creativity just comes down to putting pre-existing things together in novel and interesting ways.

WHERE TO GO FOR INSPIRATION

This is the fun part. There is a ton of information out there to glean. The trick is finding it. Here are a few gems that many designers use as their personal inspiration wells:

1. Pinterest
   (https://www.pinterest.com/)
2. Tumblr
   (https://www.tumblr.com/)
3. Awwwards
   (http://www.awwwards.com/)
4. Cool Hunting
   (https://www.coolhunting.com/)
5. Creative Bloq
   (http://www.creativebloq.com/)
6. Smashing magazine
   (https://www.smashingmagazine.com/)

An old-school but still very useful way to store inspiration is to create a mood board. A mood board is basically a large collage made up of text, pictures, and even physical objects that are related to the current project.

Mood boards are effective because having an actual physical thing in front of you that you can touch, rather than something digital on a screen, effects the brain more powerfully. Also, a mood board can’t be turned off. Whenever you need to rest your eyes or take a break, the mood board is there to draw your attention and make you consider it. In consequence, moments of boredom can sometimes turn into sudden moments of creative insight.

HOW TO STORE INSPIRATION

Storing inspiration is not easy (someone should really build an app for that). There was a time when designers would keep magazine clippings, album jackets, and various inspirational material in their desk drawers to be pulled out as needed.

These days, there are online and digital options available for storing inspiration (including some of the suggestions above). These sites often allow you to tag photos and organize them in ways that are meaningful to you, and also share or collaborate on inspiration libraries with your friends or colleagues.
FEEDING YOUR INSPIRATION ENGINE

So here’s how design thinking works. It usually happens when you aren’t paying attention. Great ideas come along in the strangest places—in the shower, during your commute, on your morning jog. It usually occurs after you’ve thought about a problem for a long time and then stopped thinking about it for a while. It’s as if your subconscious brain starts working even harder on the problem when your conscious brain takes a break.

Designers understand this well. After collecting and digesting all of their inspirational material, designers know that their real task is to relax and let the mind do its job. The essence of design thinking, therefore, is learning to alternate between observing the world carefully on the one hand, and then forgetting about it completely on the other.

Anyone who has to look at a blank app and figure out what it needs to look like can always use some inspiration. Over time, designers have come up with techniques for maximizing their chances of coming up with good ideas, and it begins with asking what, where, and how. Successful designers collect pieces of inspiration that will help them to come up with good ideas and then meditate on them. They use tools like online and physical mood boards to help them organize their visions. More importantly, however, successful designers also know when to look away.

For additional inspiration on this topic, check out:

- Bill Buxton on Design Thinking
- The Importance of Design Thinking
- Feel: A Crash Course for Architects on Design Thinking
If design can be said to have one big secret, something that makes designers what they are and distinguishes them from coders, it is sketching. At the same time, it is an easy skill to learn—though not to master—and you have probably been doing it all of your life without realizing it.

We have covered hard skills like typography and visual communication, as well as soft skills like finding inspiration. This is the first section, however, in which you will be asked to actually do something with your own hands. Sketching is a design practice rather than a design theory, and to understand it, you have to actively engage in it. In this way, it shares some things in common with developer practices like agile and test-driven development (TDD). Simply reading about these practices won’t ever tell you if they are effective or not. You can’t know if they actually work unless you try them for yourself.

**WHAT IS SKETCHING?**

Sketching has four core attributes that make it both a time saver as well as a conserver of emotional energy. Sketches are:

- Disposable
- Quick
- Timely
- Inexpensive

Pablo Picasso used to make sketches on paper napkins and pay for his meals with them. If nothing else is available, you can certainly use napkins, too—and plenty of designers will swear that nothing else will do.

There is a certain amount of pleasure to be gained from using better tools, however. For a small sum of money, you can purchase a Moleskine notebook in which to save your sketches. If you drop by an art store, you can pick out a set of thin anime pens or colored pencils. However, a standard No. 2 pencil will work just as well.

To start, get your writing surface—napkin or otherwise—and drawing instrument ready. Think of a project you are currently working on. Consider the layout, or possibly the workflow. Now draw it. Draw whatever comes to mind when you think of this project.

**SKETCHING YOUR APP DESIGN**

If design can be said to have one big secret, something that makes designers what they are and distinguishes them from coders, it is sketching. At the same time, it is an easy skill to learn—though not to master—and you have probably been doing it all of your life without realizing it.

We have covered hard skills like typography and visual communication, as well as soft skills like finding inspiration. This is the first section, however, in which you will be asked to actually do something with your own hands. Sketching is a design practice rather than a design theory, and to understand it, you have to actively engage in it. In this way, it shares some things in common with developer practices like agile and test-driven development (TDD). Simply reading about these practices won’t ever tell you if they are effective or not. You can’t know if they actually work unless you try them for yourself.
To make this exercise easier, plan to throw whatever you draw into the recycle bin once you are done. Remember: the first attribute and rule of sketches is that all sketches are disposable.

Try to sketch quickly and enjoy the simplicity of the activity. Though it may seem irrelevant at first, take a moment to note the resistance of the pencil as it scratches across the paper, or the smoothness of the pen on paper, or the way the mouse rolls across your pad if you are using a drawing program. Are there sticky bits on your mouse that make it occasionally skip? This attention to the experience of sketching is to remind yourself that it is the activity of sketching as much as the result which is important.

Are you done? Throw your sketch away and start another. The practice of sketching epitomizes the principle of fail early and fail fast. It allows you to energy into it.

In his book “Design for Software: A Playbook for Developers”, Erik Klimczak, creative director at Truth Labs in Chicago, writes “Sketching is the quickest and most disposable form of generating ideas.” Like code-first and similar coding practices, it is a way to jumpstart yourself with little effort or commitment. The benefits, however, are extraordinary.

One of the keys to good sketching is to keep your ideas loose. Chicken scratches and stick figures are fine. The drawings are just byproducts of your creative process. In fact, if you are already a good artist, you might even want to reign yourself in a bit. Once you ease into the practice, you will find that sketching lets you develop your ideas and find any potential problems early. It helps you to find alternatives and stay flexible instead of locking into one path early in your design process. Perhaps most helpful of all, it can be done at any step in the software creation workflow. Whenever you need more ideas, pull out a napkin and start sketching.

**THE PHILOSOPHY BEHIND SKETCHING**

With his 2007 book, Sketching User Experiences, Bill Buxton took the design community by storm by showing how sketching can and is being used in digital design, and more importantly for us, how it is essential for technology design, in which everything we do is—in some sense—a brand new experience.

The first-time reader of Sketching User Experiences is often surprised to find that all of the great things they’ve heard about the book are difficult to actually pin down. Buxton’s book is a meandering journey with insights into the gradual design of the iPod, the nature of creativity, some projects he’s worked on in the past, and lots and lots of examples of sketching. Along the way, his readers also learn about mountain bike design, Bill’s obsession with orange juicers, and a bit about canoeing.

This peculiar format of the book distills what it means to think like a designer. It goes down rabbit holes, pulls together apparently incongruent anecdotes, and constantly tells stories to pull readers along until the readers resolve for themselves certain insights about design thinking.

Arguably, the most important insights are these:

- That the user interface is the same thing as the user experience
- That sketching is a way of thinking
- That sketching is a way to communicate with others

**WHAT IS THE USER EXPERIENCE?**

Buxton’s favorite orange juicer is a legendary meme in the design world. He uses his various experiences with orange juicers to illustrate what it means to design a user experience. Whereas a UI is simply how a product looks, the user experience includes not only how a product feels but also the anticipation of using it and the enjoyment of it when you aren’t using it.
Translated to the app world, user experience is not just about the color theme you choose and the navigation scheme you implement, but also how your user responds to your app and feels about it. In what hand does the user hold the app? Where is the user when using your app? Is he or she with other people or alone? Sketching is a way to approach these bigger experiential questions about how your users interact with your app. Quick sketches aid the imagination and let you figure out design issues before you even have a line of code written yet.

SKETCHING AS A JOURNEY
Sketching allows us to work out our ideas without consequences, which in turn gives us a lot of freedom to wander and explore. According to Buxton, “Sketching in the broad sense, as an activity, is not just a byproduct of design. It is central to design thinking and learning.”

There is a great temptation to think of sketching as a first draft of the app or as a tool for prototyping. It is not. To think of it in this way immediately places importance on the sketches themselves, which in turn ratchets up our performance anxiety and potentially causes the paralyzing fear that besets many projects. Remember: the sketches themselves are utterly disposable. It is the act of sketching that is invaluable.

SKETCHING IS COLLABORATIVE
Sketching is the design equivalent of whiteboarding for developers.

Most developers have much more experience with whiteboarding than with sketching. Have you ever noticed how sloppy whiteboards are? There is usually no real concern with legibility even though we all pretend that whatever is written needs to be saved and will be referred to again at some future, indeterminate time. This is a peculiar lie we tell ourselves so we can use whiteboards to facilitate talking and communicating about our ideas. There’s a reason we don’t use permanent markers on white boards. That’s not what whiteboards are for.

Sketches function in a similar way. They let us think through our ideas privately and then communicate them publicly. At the same time, no importance should be placed on the physical sketches themselves.

YOUR SKETCHING CHALLENGE
The most important thing at this stage is to prove to yourself that sketching provides some personal benefit. Here’s your challenge: spend an hour and make 30 sketches. That’s about a sketch every two minutes. You may find that it opens creative doors for you and provides a mighty new tool in your arsenal for developing beautiful apps. What have you got to lose? At worst, it’s just an hour of your time.
Conclusion

Whether you are a lone developer or a member of a small development team, the skillful application of design know-how can take your app from good to great, even if there is no designer on your staff. What’s more, putting great design principles into practice at the start of your process will save you wasted coding time down the road while also resulting in a more seamless, pleasing user experience.

While this document covered mainly visual design principles, it is important to keep in mind that the world of apps is constantly changing—UI design will need to change with those shifting expectations. Emerging experiences, such as holographic computing and bots, challenge our notions of what a good UI consists of.

A bot, for instance, may have no visual UI at all. It may simply be the endpoint for a chat application or a phone text. Nevertheless, it will still require a good UI designer to help plan out the tone of the bot and how the bot can provide affordances to help the user understand how it’s used.

For this reason, development and UI design must always work hand-in-hand. Design without code is empty; code without design is blind. Coders need to let designers know what is possible in an app while designers need to help coders understand what is desirable. In order to build the best apps, each must meet the other half-way and understand what they respectively bring to the table.

The time of siloed individuals is coming to an end. The future of computing may very well belong to the hybrid—the jack-of-all-trades who is as capable with a compiler as she is with a digital paintbrush. In an ever-changing world, it behooves the modern developer to never become complacent, but instead to constantly learn new skills and move beyond his comfort zone. This whitepaper is an attempt to help you along that journey.
## Index

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Who is this whitepaper for?</td>
<td>2</td>
</tr>
<tr>
<td>How is this whitepaper designed?</td>
<td>2</td>
</tr>
<tr>
<td>Introduction</td>
<td>4</td>
</tr>
<tr>
<td>Demystifying design</td>
<td>4</td>
</tr>
<tr>
<td>Getting started</td>
<td>4</td>
</tr>
<tr>
<td>What is the focus of your app?</td>
<td>4</td>
</tr>
<tr>
<td>Understand your customers</td>
<td>5</td>
</tr>
<tr>
<td>Be inspired – horde good ideas</td>
<td>5</td>
</tr>
<tr>
<td>Fail early, fail fast: Prototypes</td>
<td>6</td>
</tr>
<tr>
<td>Validate your gut</td>
<td>7</td>
</tr>
<tr>
<td>Understanding typography and UX design</td>
<td>8</td>
</tr>
<tr>
<td>Kerning</td>
<td>8</td>
</tr>
<tr>
<td>Character width</td>
<td>9</td>
</tr>
<tr>
<td>Leading</td>
<td>10</td>
</tr>
<tr>
<td>Serif and sans-serif</td>
<td>10</td>
</tr>
<tr>
<td>Fonts (and font families)</td>
<td>11</td>
</tr>
<tr>
<td>Using iconography to enhance UX design</td>
<td>13</td>
</tr>
<tr>
<td>When to use custom iconography</td>
<td>14</td>
</tr>
<tr>
<td>Where to find icons</td>
<td>15</td>
</tr>
<tr>
<td>Adding color to your design</td>
<td>16</td>
</tr>
<tr>
<td>Color and emotion</td>
<td>16</td>
</tr>
<tr>
<td>Color theory</td>
<td>17</td>
</tr>
<tr>
<td>Color contrast and accessibility</td>
<td>18</td>
</tr>
<tr>
<td>Color schemes</td>
<td>19</td>
</tr>
<tr>
<td>Accessibility and your app design</td>
<td>20</td>
</tr>
<tr>
<td>Be accessible</td>
<td>20</td>
</tr>
<tr>
<td>Reduce UI button pressing</td>
<td>22</td>
</tr>
<tr>
<td>Not everyone has perfect vision</td>
<td>23</td>
</tr>
<tr>
<td>Scale</td>
<td>25</td>
</tr>
<tr>
<td>Communicating with your users</td>
<td>28</td>
</tr>
<tr>
<td>Friendliness</td>
<td>28</td>
</tr>
<tr>
<td>Provide a helping hand</td>
<td>31</td>
</tr>
<tr>
<td>Tone</td>
<td>32</td>
</tr>
<tr>
<td>Visual communication and visual cues</td>
<td>33</td>
</tr>
<tr>
<td>Visual cues in general</td>
<td>33</td>
</tr>
<tr>
<td>Affordances</td>
<td>35</td>
</tr>
<tr>
<td>Calls to action</td>
<td>36</td>
</tr>
<tr>
<td>Designing for Intuitive Navigation</td>
<td>38</td>
</tr>
<tr>
<td>Consistency</td>
<td>38</td>
</tr>
<tr>
<td>Simplicity</td>
<td>40</td>
</tr>
<tr>
<td>Clean interaction</td>
<td>40</td>
</tr>
<tr>
<td>Rules of Thumb</td>
<td>41</td>
</tr>
<tr>
<td>Polishing Your App with Animations and Audio Cues</td>
<td>43</td>
</tr>
<tr>
<td>Animations that call attention</td>
<td>43</td>
</tr>
<tr>
<td>How long should an animation stick around?</td>
<td>45</td>
</tr>
<tr>
<td>Animations that transition</td>
<td>45</td>
</tr>
<tr>
<td>Sound effects</td>
<td>46</td>
</tr>
<tr>
<td>Best practices for sound design</td>
<td>46</td>
</tr>
<tr>
<td>In UWP apps, audio often goes underused</td>
<td>47</td>
</tr>
<tr>
<td>Design thinking: finding your inspiration</td>
<td>48</td>
</tr>
<tr>
<td>Understanding what kind of inspiration you’re looking for</td>
<td>48</td>
</tr>
<tr>
<td>Where to go for inspiration</td>
<td>49</td>
</tr>
<tr>
<td>How to store inspiration</td>
<td>49</td>
</tr>
<tr>
<td>Feeding your inspiration engine</td>
<td>50</td>
</tr>
<tr>
<td>Sketching your app design</td>
<td>51</td>
</tr>
<tr>
<td>How to sketch</td>
<td>51</td>
</tr>
<tr>
<td>The philosophy behind sketching</td>
<td>52</td>
</tr>
<tr>
<td>What is the user experience?</td>
<td>52</td>
</tr>
<tr>
<td>Sketching as a journey</td>
<td>53</td>
</tr>
<tr>
<td>Sketching is collaborative</td>
<td>53</td>
</tr>
<tr>
<td>Your sketching challenge</td>
<td>53</td>
</tr>
<tr>
<td>Conclusion</td>
<td>54</td>
</tr>
</tbody>
</table>