

Chapter 3

Bending users to your will

Interaction design principle #3: Provide alternate paths to enlightenment



Boy: *Do not try and bend the spoon. That's impossible.
Instead... only try to realize the truth.*

Neo: *What truth?*

Boy: *There is no spoon.*

Neo: *There is no spoon?*

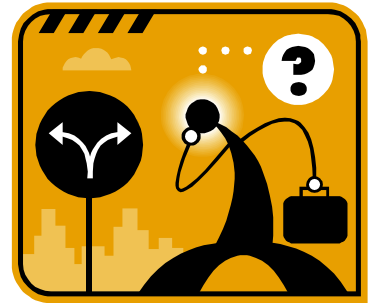
Boy: *Then you'll see, that it is not the spoon that bends,
it is only yourself.*

Why Bending Spoons?

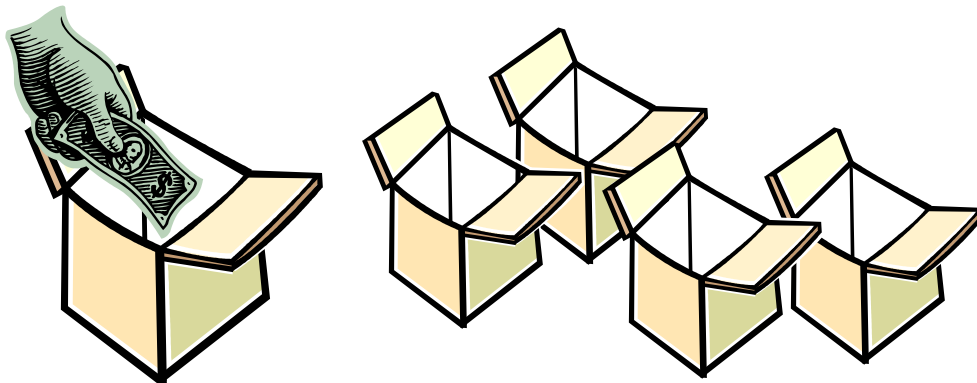
You have as much chance of forcing a user to do what you want as you have bending a metal spoon with your mind. Bending spoons is a metaphor for getting users to click where you want them to click and think the way you want them to think.

Every website and every application has features. Sometimes these features are simple and obvious. Other times the features are complicated and awkward. You, the decision maker on your project, are faced with choices about how to present functionality to the user. Your goal is to have the user complete their task quickly and enjoyably with minimal errors. This is where the spoon bending comes in.

Most applications force users to accomplish their goals in one specific way. However, users do not always know the way that works best. They rely on past experience to make an educated guess. More often than not, they are incorrect and must guess again. Users will keep guessing until one of those guesses are right or give up trying. Forcing a user to keep trying until they find the right choice is like bending their natural tendencies towards your own. The better way is to let go of your own preferred task model. The truth is that there is no best way.



To illustrate, we can imagine presenting a friend with five boxes. You have hidden \$20 in one of the boxes. Your friend is told that he must find the money.



Statistically speaking, people will find the money 20% of the time on the first try. The success rate will diminish from there because people will quit trying. Very few will guess incorrectly four times and still keep trying. The majority of people will assume that there they have been tricked.

A better approach to the problem would be to hide \$20 in every single box. That way the user will guess right 100% of the time. This is the basis of all great interaction design. Transform all of the possible guesses the user could make into valid choices to drive the task. Let the user bend the application how he or she sees fit.

Case Studies are a great way to illustrate the good and bad choices that designers make. In the following sections we look at several different tasks and how it is possible to achieve the goal.

Case Study: Gmail

An example of how a popular program can go wrong by bending users is the breakthrough web program Gmail. Gmail is a perfect demonstration of how some things they got right overwhelm the things they did wrong. If they were to unbend on some

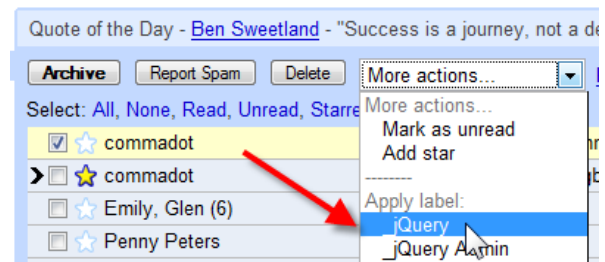
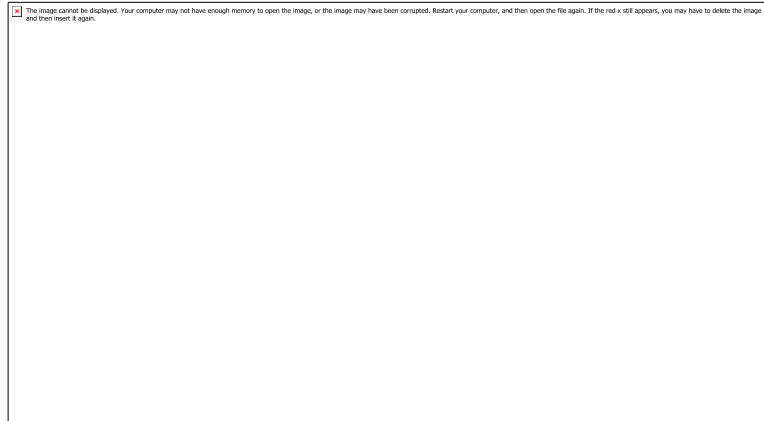
interface options, then task completion for those features would rise. However, their popularity stems from other features in the application. Take a look at how you have to move a message out of your inbox into a labeled area.

The most obvious is to drag and drop the message into the label. That doesn't work. No keyboard shortcuts either. I can't right click on the message to do it. No icon available to click on. In fact, there is no indication of how to do this very common task. The answer is hidden in the "more actions" select box.

So, of the five methods they could have chosen, Gmail provides exactly one way. All of my guesses failed until I found the right

choice. I have been bent to the will of the Gmail designers. I have learned their "best and only way" because I like the program overall and I am willing to put up with it.

Task completion is something to constantly test and keep in perspective. The overall mission of a program is to be popular and useful. Some believe that a single path to task completion makes Q&A and support easier. Although this is true, it still is desirable to have alternate methods of getting the job done. As the web matures, we will see richer and more diverse interfaces.



The user does not need to know that alternative choices exist. They are perfectly content with the ways they have discovered, even if that they only find one. Do not feel the need to explain the choices to the user. As long as you have put the \$20 in each box, then the user can not go wrong.

Case Study: Printing

The five boxes represent the major categories of User Interface for actions. To help demonstrate, we can look at “Printing” as a case study on how to provide multiple paths to successful task completion.


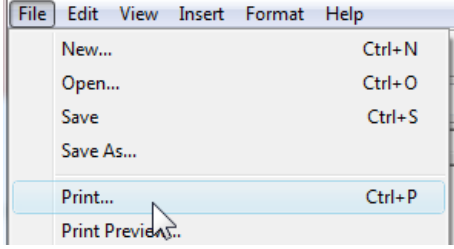
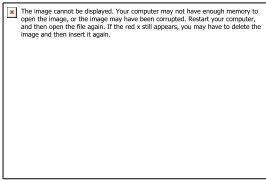
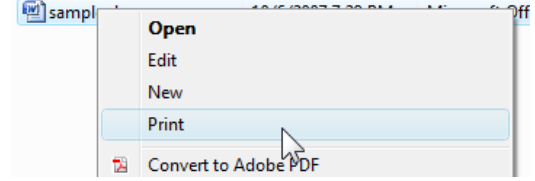
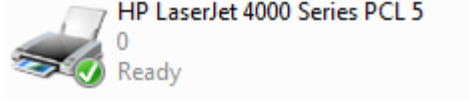
<p>Icons. Small graphic that looks like a printer, usually in a taskbar at the top of the screen.</p>	
<p>Menus. Often grouped at the top left of the application. The most common placement for print is in the first menu because it is such a frequent action.</p>	
<p>Keyboard shortcuts. Control-P and F12 are common printing shortcuts. Menus can also be accessed via keyboard shortcuts too. Menus are usually accessed with the ALT or Option Key.</p>	
<p>Context Menus. Right clicking will result in different menus depending on what you click on. Right click on the document file and you get a print option.</p>	
<p>Drag and Drop. Although it is the least used, you can also drag a file onto a printer icon, which will cause the document to print.</p>	

Table 3.1: The five major user interface methods of task completion

Printing is an example of great interaction design. No matter how crazy of a plan the user has to print, most likely it will work. Users will try everything. In this case, there are five different ways to print the same document. A user will find at least one correct way to print on their first try. Without having to guess wrong a user will feel confident and productive.

By giving multiple paths to task completion, you are helping the user find their functionality in the first place they look, and allow them to stop looking. These alternate methods do not program themselves into your application. They take time and you have to fight for them in the designs otherwise they will be cut by management and the project managers.

The payoff is higher task completion rates and overall satisfaction. Of course, to know this you will have to measure the success of your site. As the saying goes, "If you can't measure it, you can't manage it."

Why do you always recover a lost item in the last place you look?

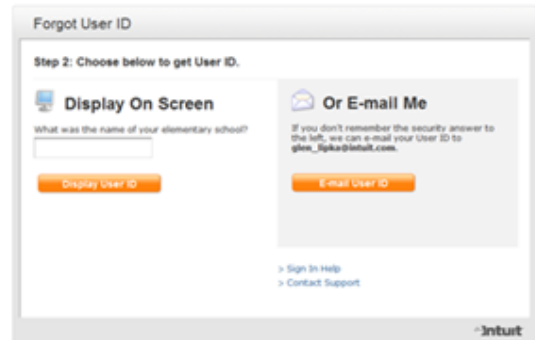
Because after you find it, you stop looking.

--- Old Joke

Case Study: Forgot Password

One example of how alternate task completion methods can pay off is when I redesigned a Fortune 1000 "Forgot Your Password?" feature. The site had hundreds of thousands of people a year using this function on an e-commerce site. The company did a good job of tracking success. Unfortunately, the stats said it was an abysmal 10% success rate. 90% of users gave up when it wasn't working and presumably bought their products somewhere else.

I redesigned the workflow with the principle of giving the user more than one choice of how they could recover their password. With a few weeks of effort, the new functionality improved success rate by three fold. This translated into a significant increase in sales and a drop in customer support phone calls.

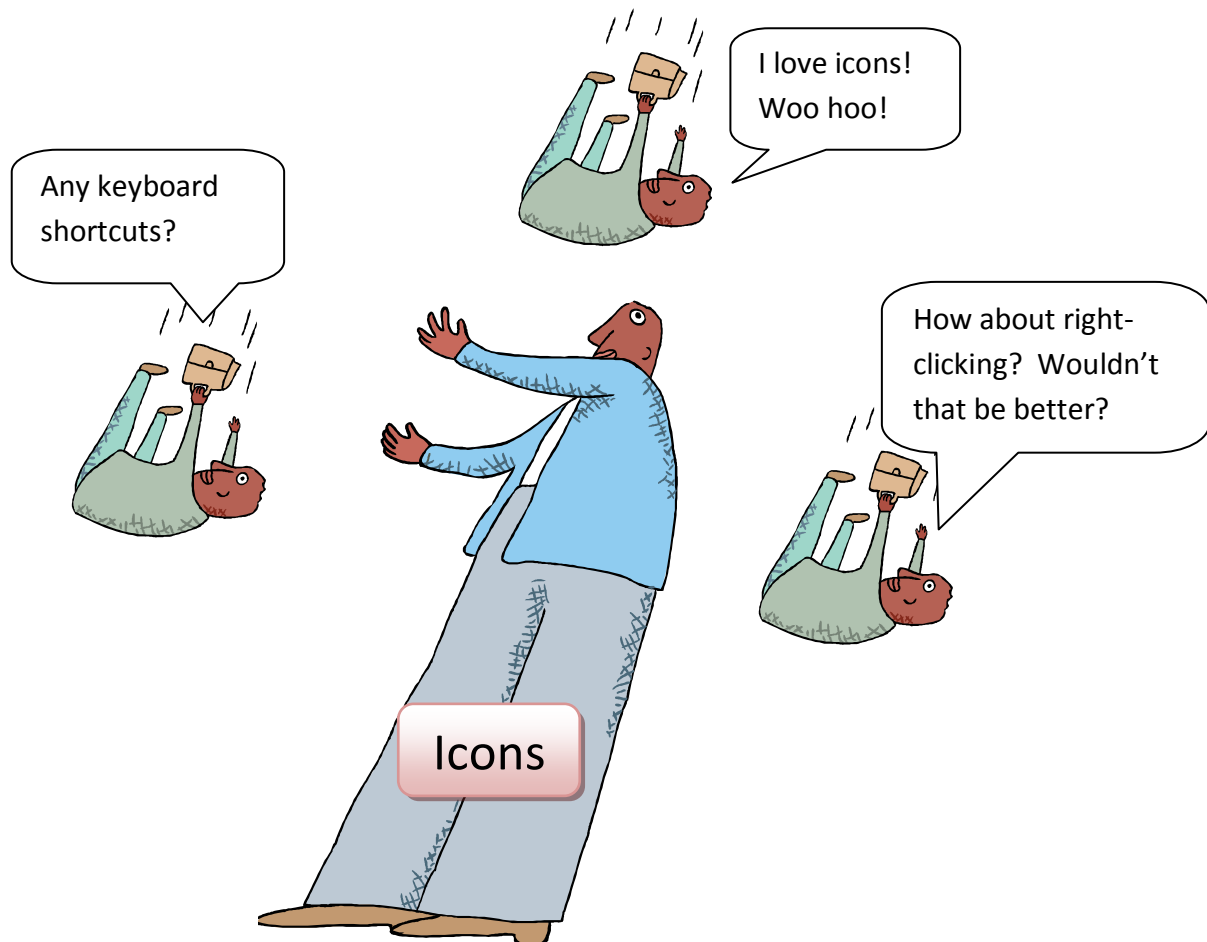


As a user experience designer you have to you have to advocate and champion each improvement. You need to educate your peers and evangelize design principles. Tangible statistics are the keys to proving success, but it's easy to test incremental or conservative changes. My original plan was to make the whole process easier for the user. In this case, the security department of the company had a high bar to meet security standards. This had the side-effect of making the task more complicated.

The five categories of User Interface are key tools for you to use to help your users. When you think about how to enable all of the possible guesses a user may make, these five are a great start. Always look for other possibilities as well. Additionally, you don't have to make it perfect on the first try. If at first you don't succeed...try, try again.

The UI Five

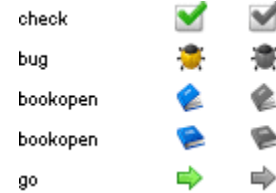
The Big Five is a family of task completion modes. It is important to have ALL of these modes available to the user. Most project teams will feel the temptation to cut up to four of these modes for the sake of project schedule. It is best to think of your users as small cartoon people falling helplessly through the air. It is your application that has to catch them. If you don't catch them, they will fall (splat!) and take their business elsewhere.



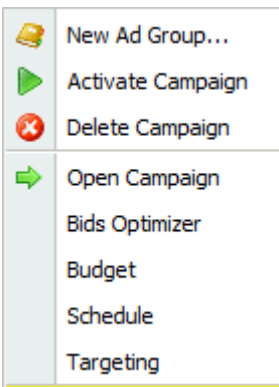
Some engineers believe that users are being illogical and by providing only a single method of completing the task, you will be training them to think logically. You see, they are performing a public service. They are helping us “backwards folk” be more logical. It’s just like that episode on Star Trek: The Next Generation, where the Picard-Borg said, “You will be assimilated. Resistance is futile.”

Remember, there will be users who try to use these modes whether you provide them or not. A polite and lovable application is one that allows the user to use whichever mode they prefer. Although we will go into detail on these later in the book, it is useful to get an introduction.

Icons. *A picture is worth a thousand words.* A simple printer icon will give plenty of indication on what it does and it won't take up much space. Additionally, always use icons inside menus to help the user scan and find the action they are looking for.



Icons also breathe life into your site or application. Their little splash of color will brighten up your application. Details about icons including sizes and how to implement are covered in chapter 4, The UI Five in Detail.

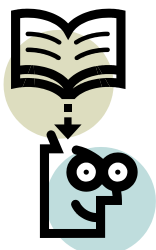


Menus. *Sometimes a word is worth a thousand pictures.* Sometimes the action you are trying to provide just does not break down into a neat 16x16 icon. Use words with your icons when this is the case and provide consistent menus to organize your actions. If your menus are consistent then your application will be scalable and easy to use.

Menus are very valuable tools to save space and organize functionality or navigation, but they have many pitfalls. Details about menus are also in chapter 4.

Keyboard Shortcuts. Some people are mouse people and some people are keyboard people. I happen to be a keyboard person. I press control and b on my keyboard to **make this bold**. Someone else might press ALT and find the proper item in a menu, using nothing but the keyboard. Keyboard shortcuts are very often left out of applications. This is a great opportunity to get “unexpected WOW”* out of your customers. You will allow users to become “power users” when they discover these keyboard shortcuts. Plus, visually, your application requires no additional screen real estate to add in keyboard functionality.

Context menus. Also known as right-clicking, a context menu is incredibly powerful for the user. One of the most powerful ways context menus can enable the user is in an explorer tree. A tree is a very popular way of navigating a large application and the ability to right-click on tree items is critical to creating some fast-track paths for the user. Users will love you for every right-click menu that you add to tree nodes. Make sure to do the *right* thing!



* Definition: **Unexpected WOW!**

[uhn-ik-spek-tid wou] –Noun

The smile and bright-eyed expression on a user's face when something exceeds their expectations.

All of these modes of user interface have a dark side. They can be overused or misused. The following case study demonstrates how right-clicking in only one spot of the app could lead to trouble.

Case Study: New Document

There are rules and implications for context menus. Especially around what should be right-clickable and what shouldn't. Implementing these UI elements incorrectly will do more harm than good to your application. User experience can be terrible with the overuse or misuse of UI elements.

In one application I was designing, an engineer added a context menu to a small part of the application. We had not yet built context menus throughout the app so this was a new innovation. Some users discovered (or were told) about the new feature. Despite the fact that those users enjoyed the ability to right-click in that one area, we experienced negative side-effects. All of our users who discovered that feature started right clicking on every item in every section of the app. These users started to complain that right-clicking wasn't available everywhere.

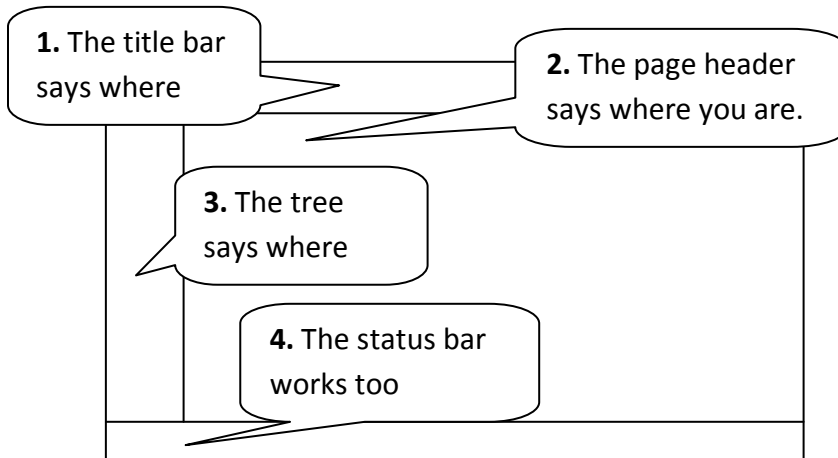
On the other hand, because we didn't use right-clicking consistently, most users never discovered this feature. It is important to think about the overall model of how things work in your application and introduce improvements in a way that will be discoverable and not mismanage expectations.

In addition to the five major UI components there are two techniques that should be employed to give the best user experience possible. These are "Repetitive Navigation" and "Omni-directional Task Completion". Using these two techniques will provide a safety net, through which none of your users will fall. It will also make your application easy to use and easy to learn.

Repetitive Navigation uses multiple navigation options to know where the user is and how to go somewhere else. Omni-directional task completion breaks the task down into building blocks and lets the user go in any order they want.

Repetitive Navigation

The most common problem in an application is the user losing track of where they are. I have seen many people dismiss this problem, but it is absolutely critical. It's not enough to put one indication on the page. You should put as many road signs as possible. Those familiar with the application may claim the indications are redundant or even repetitive, or possibly redundant. However, a new user will feel comfortable with the added clarity.



In general, people do not read web pages or web applications. They scan the page in a random and chaotic manner. Actually, it's not random or chaotic at all. Eye tracking technology has allowed researchers to understand the pattern of people's eyes on a computer screen. The same rules that applied to 16th century painters still applies today. Composition, balance and white-space play a critical role in helping the user find the useful navigation that you have placed so carefully on the page.

In 2006, Jakob Nielsen* conducted an eye tracking study which showed a clear F or E pattern on many websites. This means the user scanned the page quickly, looking on the left first, and then in brief horizontal motions. This knowledge helps the designer place the indications and navigation in the proper places.



I like to think of users as very intelligent, as well as, very distracted people. Maybe even people with little children or co-workers running around. I imagine my users are multi-tasking like crazy when they are on my site or application. Imagine your users are intelligent human beings during Halloween trying to use your application when the doorbell rings every two minutes. In this sort of world, you have to be persistent.

*Source: http://www.useit.com/alertbox/reading_pattern.html

For instance, you have to constantly remind the user where they are, what they are doing and what they should read. Knowing that you only have a brief moment to catch their attention, you have to be clear and concise. Knowing that people scan information and ignore instructions, you have to put information in multiple places in a variety of styles. Do not bend the user to your will by making them look in one and only one place for the information. Make information repetitive so that, no matter where they look, they will get the message. Use every method at your disposal including titles, headers, trees, cookie crumbs, status bars, and other UI elements to increase task completion.

Be sure to make information repetitive so that, no matter where they look, they will get the message.

Omni-directional task completion

Omni-directional task completion is not as simple as just adding navigational clues. It is a broad application philosophy and not just a single element. Like other ways of bending users to the system's will, this has to do with the possible chronological order that a user can complete a complex task. Whether it is an e-commerce website or a workflow application, there are diagrams that get produced which show how the user is supposed to go about their task.

These diagrams are written in a logical and clear fashion. They show how a user can complete their task with the minimum of errors. The unfortunate aspect of these diagrams is the assumption that the order they go in is the only one available. The question often arises, "What if they do Step 3 first?" The answer to this problem has traditionally been The Wizard.



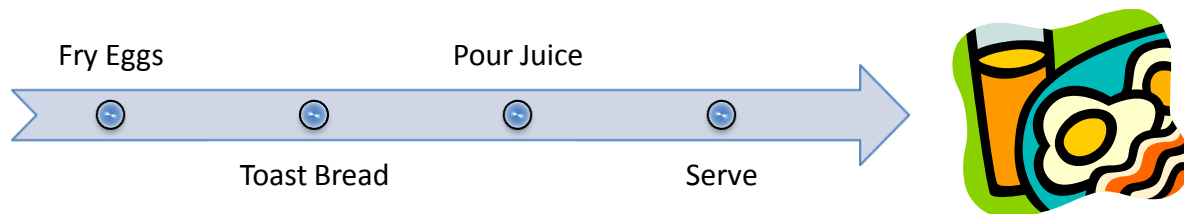
The Wizard was invented to stop users from going in the wrong order. They are like the "It's a Small World After All" ride in Disney Land. You are on tracks and can only move one painful step at a time. The wizard usually gives a next, back and cancel button. Most e-commerce sites force you to add your billing information first before your credit card information. Wizards keep the flow of the application clear and unambiguous. They also bend the user to the will of the designer and engineer. They do not allow user driven task completion.



Not all case studies need to be websites and technology. Every task that can be conceived can be micro-managed. A great analogy is worth at least three real world examples. The following case study is an average analogy, so it's only worth two.

Case Study: Breakfast

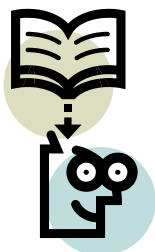
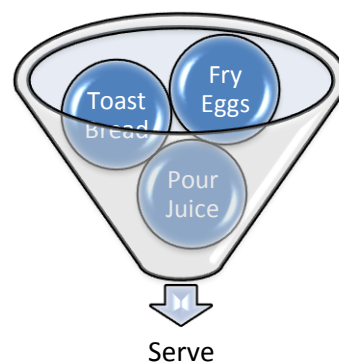
My small children are interested in cooking. I realized that the order you apply ingredients to your dish is important in many cases. However, it is not critical in every case. I started to deconstruct my breakfast to see where I could leave the order up to the individual cook and where I needed to be strict.



The straight and narrow path of task completion has important benefits. It is predictable and easy to measure and test. Don't underestimate how valuable these are to management and the analytics folks. However, they are not valuable to the user. Inevitably, the user will want, as they often do, to re-arrange the order that they do things in. My kid/chef Jared requested to make the juice before the toast. He explained that Juice starts with J and it should come first.

Although that seems like a fairly normal breakfast request, I am always surprised how often computer applications do not allow the user to go out of order. This is also known as "Wizarding the User to Death."

A more natural and lovable way to model your application is to allow users to build the objects that make up the final object in whatever order they want to go in. This means scrapping the Wizard model and taking a more "object-oriented" approach. The user would build each object separately and then combine them together when the time is right.



* Definition: **Wizarding to Death**

[wiz-erd-ing too deth] –Verb

Like death by chocolate, except with more Next, Back and Cancel buttons.

Of course, the wizard is sometimes the only logical way to get the job done. Every UI tool, including the wizard, has its place. The key is to be creative and come up with ways that the user could build reusable objects and put them together to make the result, rather than walking them down a straight and narrow path

Summary

It's all about how you view your audience. Most applications are designed for the engineers, or the managers or the designers or the user's boss or even a caricature of real user. It's designed for everyone except the actual real live human beings. Actual users are very distracted people. They are intelligent, but have been trained by a myriad of applications on how to perform certain tasks. They learn these patterns and try to apply them to your application. Additionally, they are interrupted by meetings, phones, and kids screaming "trick or treat!"

In the real world, users have seen all kinds of user interfaces and have struggled to learn how to complete their task in each one of them. There is no universal pattern that is followed. It is chaotic and random as far as the user can see. When they meet your application they have preconceived ideas about how it works. They will guess every single way to use it. For each incorrect response, they get more and more depressed, frustrated and guilty. For each success, they feel confident and content.

By giving the user multiple paths towards completing their goal, you are reading their minds. You are putting the functionality in the "first" box they look. The best user interface is invisible and auto-magical. It reads the users mind and anticipates their every guess. The only way to achieve this is to use all the tools in your toolkit and strategically place functionality in every conceivable place the user may look.

The reward for this design is higher task completion rates, lower abandonment and higher retention of customers. Not to mention the best reward of all. You will make happier people. In the following chapters we will explore how to use this toolkit effectively. Now that we know what we are aiming for, we need to master the details of user interface elements in the following chapter.

