



Practical UI Patterns for Design Systems

Fast-Track Interaction Design for a
Seamless User Experience

Diana MacDonald

Apress®

Practical UI Patterns for Design Systems

**Fast-Track Interaction Design
for a Seamless User Experience**

Diana MacDonald

Apress®

Practical UI Patterns for Design Systems

Diana MacDonald
Victoria, VIC, Australia

ISBN-13 (pbk): 978-1-4842-4937-6

ISBN-13 (electronic): 978-1-4842-4938-3

<https://doi.org/10.1007/978-1-4842-4938-3>

Copyright © 2019 by Diana MacDonald

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress media LLC: Welmoed Spahr
Acquisitions Editor: Louise Corrigan
Development Editor: James Markham
Coordinating Editor: Nancy Chen

Cover designed by eStudioCalamar

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a Delaware corporation.

For information on translations, please e-mail rights@apress.com, or visit <http://www.apress.com/rights-permissions>.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at <http://www.apress.com/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/978-1-4842-4937-6. For more detailed information, please visit <http://www.apress.com/source-code>.

Printed on acid-free paper

*In loving memory of Phillip MacDonald. Thanks for
introducing me to the wild world of wordsmithing.*

Table of Contents

About the Author	xiii
About the Technical Reviewer	xv
Acknowledgments	xvii
Introduction	xix
 Chapter 1: Introducing UI patterns	 1
What's a UI pattern?.....	1
Elements of a UI pattern	6
Other pattern details.....	9
Why care about patterns?.....	11
Design efficiently.....	11
Consistency and familiarity	12
Consistency and reuse	14
Communicating decisions	15
Communicating within teams and tools	16
Evidence-based solutions	18
Context-specific, tailored solutions	19
Content contributors without a web design background	19
Learning from the experts	19
Learning how to improve experiences from patterns.....	20
Summary.....	21

TABLE OF CONTENTS

Chapter 2: Tap into patterns	23
Learning new patterns (sharpening the saw)	23
Pattern collections.....	24
Pattern galleries	25
Platform guidelines	26
UI frameworks	28
Pattern libraries, design systems, style guides, and anti-patterns	29
Inspiring visual style.....	29
Pattern: Walkthrough	30
Pattern: Playthrough.....	37
Pattern: Newsletter signup	43
Pattern: Validation feedback.....	48
Identifying relevant patterns	53
Searching for patterns	54
Pattern: Social signup.....	56
Pattern: Lazy signup	57
Competitive analysis.....	58
Learn from the best	60
Pattern: Notifications	61
Pattern: Good defaults	68
Pattern: Coachmarks	69
Implementing and tailoring patterns.....	73
Pattern: Progressive disclosure	74
Pattern: Staged disclosure	75
Pattern: Progressive reduction	79

Bringing it all together	81
Pattern: Login form.....	81
Example: Redesigning a login form	83
Summary.....	91
Chapter 3: Deciding which pattern to use and when	93
Context.....	93
Pattern: Autocomplete.....	94
Autosuggest.....	96
User context and performance	98
Pattern: Search filters	99
Information architecture	102
Multiple filters	104
Filter feedback.....	105
Choosing filters.....	107
Live filters.....	108
Batch filters.....	109
Error prevention and recovery	111
Pattern: Activity feed.....	112
Pattern: Favorites.....	115
A rose by any other name.....	117
Microcopy	119
Pattern: Pagination.....	120
Where to draw the line	121
Accessibility	122
Pattern: Infinite scroll.....	123
Principle of choices in action	125
Principle of disclosure in action	126

TABLE OF CONTENTS

Inclusive design	127
Infinite scroll vs. pagination	128
Infinite scroll and favorites	128
Pattern: Follow	129
Pattern: Friend's list	131
Friends and followers	132
Interaction and motion design	133
Triggers	134
Action	136
Feedback	136
When is a pattern a bad idea?	137
Example: Login form	137
Summary	142
Chapter 4: Patterns in design systems	143
What's in a name? The devil is in the details	143
Pattern libraries	144
Design systems	144
Related design, code, and content resources	145
Editorial style guides	145
Brand guides	147
Design guidelines and visual language	149
Style guides relating to code	151
Templates and Content Management Systems (CMSs)	158
Building design systems using patterns	158
When to use a design system	159
Framing	160
Workflows and design processes	162
Pattern previews	164

Code assets.....	176
Prototyping in the browser.....	176
Writing code	177
Converting design elements to code	178
Design assets.....	180
Writing and content.....	182
Documenting patterns or components.....	182
Extra design system features	183
Growing a design system.....	186
Kick off	186
Assembly	187
Versioning.....	188
Serving assets.....	191
Tools	191
Evolution.....	192
Summary.....	192
Chapter 5: Anti-patterns and dark patterns	193
What are anti-patterns?	193
Why care about anti-patterns?	195
Anti-pattern: Hamburger basement.....	196
What are dark patterns?.....	202
Manipulinks and Confirmshamers	203
Design smells	206
Too Much Information (TMI).....	207

TABLE OF CONTENTS

Modals and pop-ups209

 Modal UI pattern209

 Modal design smell213

 Pop-up anti-pattern214

 “Overall pattern” design smell219

The lifetime of a bad pattern.....220

Chapter 6: Mixing and matching patterns223

 How to combine patterns successfully to build a more complex UI:

 Scoped searches example224

 Reuse elements across patterns: Categories as search filters225

 Cut duplicate content from combined patterns: Categories as search terms231

 Efficiently combine patterns to avoid the need for others: Autosuggest and thumbnails233

 Interstitial patterns: Autosuggest and navigable categories236

 Visually combine and distinguish patterns: Categories in tabbed navigation...243

 Preserve or discard data in repeated use of patterns: Clearing filters on new searches.....249

 Clarify repeated patterns: Inline tags.....252

 Evaluate resulting trade-offs: Infinite scroll253

 Other forms of scoped search.....256

 Favoriting becomes wish listing256

 Wish lists/wish listing.....257

 Combining wish listing and lazy signup259

 When and how to break away from patterns260

 Investigate design smells260

 Review pattern principles for identified usability problems261

 Review problem and context.....262

TABLE OF CONTENTS

Strive for predictability	263
Innovate	264
How to break the rules	265
When to break patterns in design systems	267
Summary.....	268
Chapter 7: Conclusion.....	271
Looking to the future.....	272
Appendix: Suggested reading.....	275
Chapter 2	275
Chapter 3	280
Chapter 4	281
Chapter 5	285
Index.....	287

About the Author



Diana MacDonald is a Melbourne product designer, raised in the tropical north of Australia. She has relished the tech industry for over a decade, exploring the digital space with progressive organizations like Culture Amp, Bellroy, and SitePoint. At Culture Amp, she led the new design systems team to accelerate UI design and development. She values inclusive and remarkable stories. You can find out more about her on her LinkedIn profile (www.linkedin.com/in/diana-macdonald-didoesdigital) and contact her via Twitter (<https://twitter.com/didoesdigital>).

About the Technical Reviewer



Katherine Joyce is a passionate designer and developer with over 7 years of experience having worked across the financial and government sectors. She creates innovative, intuitive customer experiences and is an advocate of accessible design. As Lead UX/UI Designer at Alt Labs, she is leading the UX vision and crafting beautiful solutions driven by user needs. In her previous role she worked as a Senior UX/UI Designer for

Accenture, promoting accessible design in government services and helping automate legacy processes to improve the customer journey. She has also spent over 5 years with AXA Insurance as an Application Support Software Developer where she fixed bugs in legacy financial systems, debugged issues with browser compatibility, and suggested improvements to customer-facing journeys. She is passionate about advocating accessible design and mentoring those who would like to have a career in design or development.

Acknowledgments

Thanks to all the fabulous people involved in bringing this book to life, including the friends and associates who encouraged me and gave me feedback. Thanks to Simon Mackie and Darin Dimitroff in helping me start this book. Thanks to Wesley Moore, who was with me through all of it.

Special thanks to the folks at Apress who made this book possible.

Introduction

Right now, design systems are flourishing, evolving. Each product or web site is no longer crafted in isolation, but as part of a larger conversation, in a social web, among chatbots and machine learning. It is our duty to ensure every piece of the system speaks to each other, from the components to the people, fluently and eloquently. Thanks to the shift toward “modular design,” we see harmonious experiences composed from the ground up of independent modules, such as videos or articles that stand well on their own and yet can also be arranged to fit cohesively into a larger whole. This is only possible when we use a consistent language for designing modules in a system so that every part delights and the whole resonates.

Peacock feathers consist of fractal patterns, as you can see in Figure 0-1.



Figure 0-1. *Photo of a peacock displaying its train*

INTRODUCTION

Here you can see the beauty of the peacock's feathers emerges from the repeated pattern presented together.

My hope is for this book to serve as a guide to designers and makers toward strong foundations upon which we build. With a solid grounding, we can spend more time remixing our ideas into tailored and personalized experiences. We can spend more time exploring the cutting edge, innovating, and crafting beautiful user experiences.

Who should read this book

This book is for designers, developers, marketers, and makers familiar with the basics of building the Web who want to produce better user experiences in digital products. This book will help you learn how to discern good from bad, build on existing communities of practice, and dig deep into fundamentals.

What you'll find in this book

This book offers a concise guide to UI patterns: the tested, proven general mechanisms for solving recurring user interface problems, so that you don't have to reinvent the wheel and can instead focus on designing solutions to the unique problems in your business.

You'll find a smattering of code samples or visual examples throughout the book—only as much as is needed to demonstrate the idea and get you started. You'll also find

- Methods for discovering, evaluating, and implementing patterns according to best practices
- Specific examples of real-world, business-critical UI patterns, including onboarding new users, information seeking and social sharing, as well as e-commerce purchase journeys

- Vocabulary to help you match solutions to problems
- Overview of the digital landscape and resources for further learning

Chapters 2 and 3 explore user signup and onboarding to highlight the process of discovering, evaluating, and understanding patterns through the theme of finding, reading, and sharing information.

Chapters 4 and 5 will cover how to consistently apply solid patterns through design systems and pattern libraries and how to avoid anti-patterns.

Finally, we'll explore mixing and matching patterns for e-commerce in depth. This is where the magic happens. Most of the book will focus on potential solutions, so you can choose the right tool for your problems. This chapter, however, will explore a specific problem space and apply pattern solutions to these problems.

What you won't find in this book

In the digital space, there's a lot of crossover among software engineering, visual design, and information architecture. While these fields have their own rich heritage and history behind them that can inform digital interactions, we'll be avoiding them because that's far too much to cover in one book. However, I strongly encourage everyone to pursue information elsewhere to learn the history of their field of practice as well as the relevant disciplines that came before. For example, while copywriters and content strategists on the Web face uniquely digital challenges writing blogs, RSS feeds, tweets, and cross-channel content, they can learn immense amounts from journalists, librarians, and traditional marketers as some challenges remain the same as those found in these preexisting fields. I'll touch on some copywriting practices for UI later on, but if you spend most of your day writing, you'll want to dig further into these fields.

INTRODUCTION

This book won't cover exhaustive lists of available patterns for all scenarios. There is no complete code library or complete collection of design assets. I will only mention some particular resources, pattern libraries, and showcases that will help you find further extensive collections of patterns for different contexts. I'll also point out common names for patterns that differ across libraries.

How to use this book

Patterns are neat because they elegantly package up all the things you need to know about interface and interaction design. They help you grow into the field. They are useful resources that you can refer back to (you don't need to memorize all of them from the start). If you have the general gist of patterns available and instant access to all of them, all you need to do is look up the pattern you need when it comes time to solve a problem. If you know a handful of relevant, similar patterns, you can look them all up and weigh them one by one.

Use this book to learn how to recognize traits across seemingly unrelated patterns and how they similarly solve problems (e.g., continuous scrolling, tabs, and pagination might have more in common than you think). Then you can start with a problem space (whether driven by stakeholder engagement, user research, or technology), clarify the problem according to real user needs, validate them, then translate those needs into pattern solutions.

Use this book to learn how to communicate UI design solutions. Most of this book will be more valuable to people new to the practice of web and product design as they develop their vocabulary to discuss interface patterns. It may, however, also help experts learn how to share their solutions as they learn how to mentor.

This book will show you concrete examples of how to discover UI patterns, evaluate patterns, and communicate solutions to design problems and user needs. Use this book as a starting point for your journey into making digital user interfaces.

CHAPTER 1

Introducing UI patterns

To help you create intuitive products, this chapter will introduce UI patterns and highlight why they're important and valuable.

What's a UI pattern?

A **pattern** is a recurring solution to a problem in a context.

I like to think of patterns as models: a pattern has a structure and can be easily used to help you solve a problem faster than building from scratch. They have a consistent and recognizable form, as well as a method of being referenced, such as a memorable name. In knitting, you might choose a pattern from a book to help you make a sweater with good sleeves, noting that some sweaters are more ornamentally complex than others. In origami, you might use a folding pattern to produce a complex sculpture from basic origami folds, such as the *orizuru* (折鶴) or paper crane shown in Figure 1-1.



Figure 1-1. *Photo of paper cranes by Rebecca Freeman*

While you might create an *orizuru* using a proven solution, there are many ways to fold a paper crane with varying levels of ornamentation, like a flapping crane or consecutive cranes.

UI patterns (user interface patterns) are found in the digital sphere of web sites, applications, native mobile apps, and other software or devices. They provide a language for discussing interactive design. They suggest function, interaction, and intent. UI patterns document reusable parts of an interface that share a purpose.

To understand UI patterns (and how they differ from components), let's explore some ideas from the UI framework, Bootstrap. First, we'll look at the thumbnails component (<https://getbootstrap.com/docs/3.4/components/#thumbnails>) from version 3 of the framework, as shown in Figure 1-2, before circling back to what makes a UI pattern.

Default example

By default, Bootstrap's thumbnails are designed to showcase linked images with minimal required markup.



Figure 1-2. Screenshot of Bootstrap 3's thumbnails component default example

The **thumbnails** pattern presents small image previews in a collection where each image is linked to a larger resource, such as a high-resolution version of the image. If the thumbnail is a preview of a product, it will link to the product detail page. If it is a thumbnail of a video, it will link to the video player to watch the video. The key features of the thumbnails pattern are as follows:

- Small images.
- Linked resources.
- It represents a collection.

You’ll frequently find images in this pattern shown alongside a title or description, as shown in Figure 1-3.

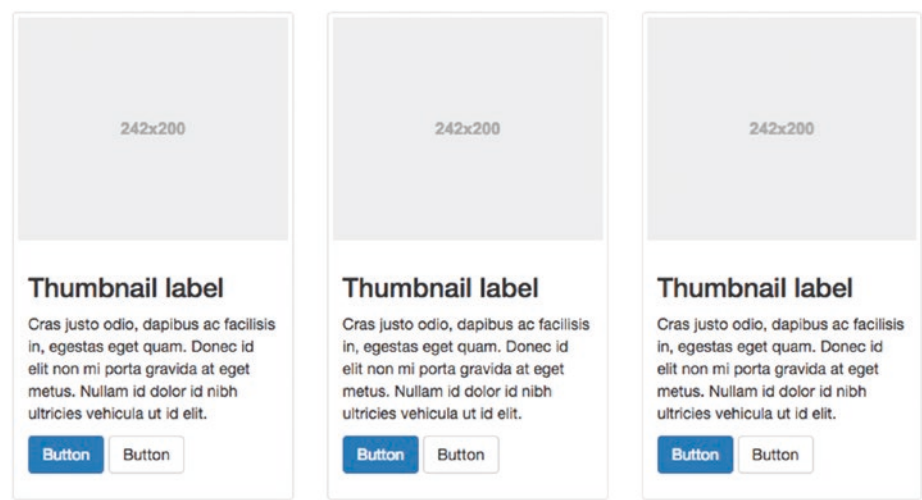


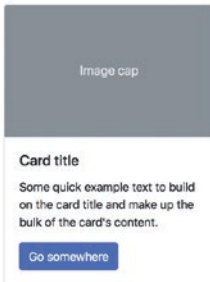
Figure 1-3. Screenshot of Bootstrap 3’s thumbnails component with custom content

In Bootstrap 4, however, you’ll find that this component’s been replaced by the *card* component (<https://getbootstrap.com/docs/4.3/components/card/>) using an image, title, text, and link, as shown in Figure 1-4.

Example

Cards are built with as little markup and styles as possible, but still manage to deliver a ton of control and customization. Built with flexbox, they offer easy alignment and mix well with other Bootstrap components. They have no `margin` by default, so use [spacing utilities](#) as needed.

Below is an example of a basic card with mixed content and a fixed width. Cards have no fixed width to start, so they'll naturally fill the full width of its parent element. This is easily customized with our various [sizing options](#).



```
<div class="card" style="width: 18rem;">
  
  <div class="card-body">
    <h5 class="card-title">Card title</h5>
    <p class="card-text">Some quick example text to build on the card title and make up the bulk of the card's content.</p>
    <a href="#" class="btn btn-primary">Go somewhere</a>
  </div>
</div>
```

Copy

Figure 1-4. Screenshot of Bootstrap 4's card component

This change more clearly separates the thumbnail's purpose of previewing visual content—a *content pattern*—from the card layout's purpose of segmenting content using repeating containers for images and text—a *display pattern*. The shift toward the display pattern makes sense for a flexible UI framework with built components. We'll learn more about components in Chapter 4.

UI patterns are more abstract than visual style. While patterns can often be identified by visual similarity, these components demonstrate it's not always so easy: a pattern describes behavior, which can be divorced from easily identifiable visual presentation. You can, for example, apply a strong, dramatic visual style or a subtle, muted flavor to a thumbnail collection.

Note You might also see reference to **user flow patterns** or **strategic patterns** when a UI pattern spans multiple pages, like in the lazy signup pattern in Chapter 2. Similarly, you might read about **behavioral patterns**, **persuasive patterns**, or **social patterns**, where the characteristic behavior presents information, shares a message, or persuades a human, like in the good defaults pattern in Chapter 2.

Elements of a UI pattern

A UI pattern is defined by three ingredients:

- A **named solution** describing *what* the pattern does
- The **problem** the user is facing or *why* this pattern is needed
- The **context** for *when* to use the pattern

For our thumbnail example

- The named solution “thumbnails” suggests a collection of small image previews linked to larger resources.
- The user’s problem is navigating a large collection of content and selecting only the items they want.
- The context is when the user needs a preview before deciding—before downloading a large file or committing to watching an entire movie. You’ll often find thumbnails on product range pages or search result listings before you’ve decided which item to drill in on. In contrast, product or detail pages need fewer thumbnails because that product or item is what you came to see so they can be shown in full without a thumbnail.

As you can see in Figure 1-5, Pinterest uses thumbnails in their visual discovery product.

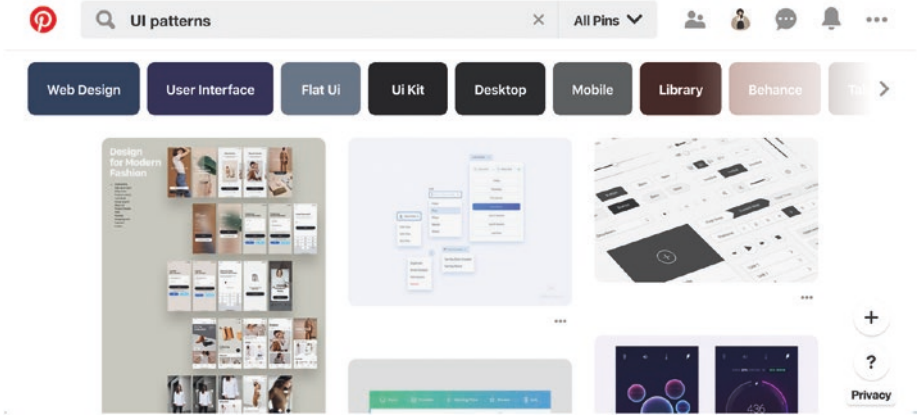


Figure 1-5. Screenshot of Pinterest thumbnails

If Pinterest continuously loaded high-resolution images at their full size instead of thumbnails, that would slow down an otherwise immersive experience. Pinterest needs to present thumbnails to facilitate smooth browsing to help people discover ideas.

In a large collection like that, you won't know what image will appear next or if it's something you want to see in detail. By using thumbnails, you can quickly browse a larger set of choices before zooming in on particularly interesting items.

Figure 1-6 shows an Adidas product with thumbnails.

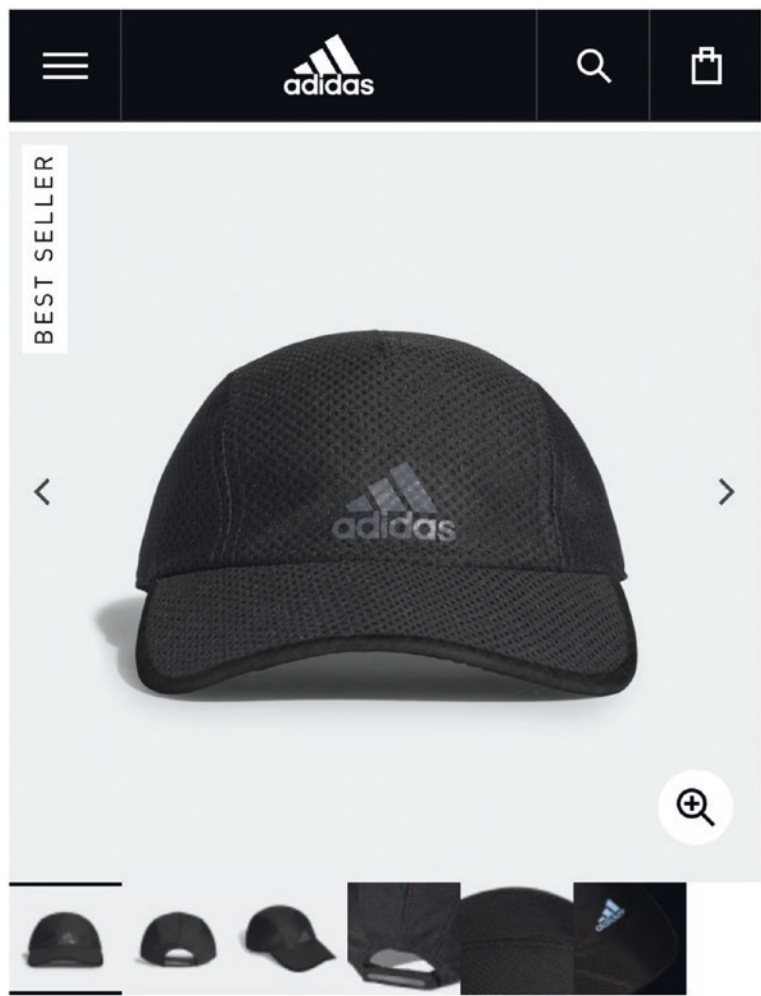


Figure 1-6. Screenshot of Adidas’s product images with thumbnails

Each small image (thumbnail) in this collection shows a preview of a product photo, linking to a larger photo. Unlike the Pinterest example, here the thumbnails are presented at the same time as the linked item.

The selected thumbnail is indicated by different styling (black borders above and below), while the larger photo is shown.

Warby Parker glasses, on the other hand, need no thumbnails to flick between product images, as shown in Figure 1-7.

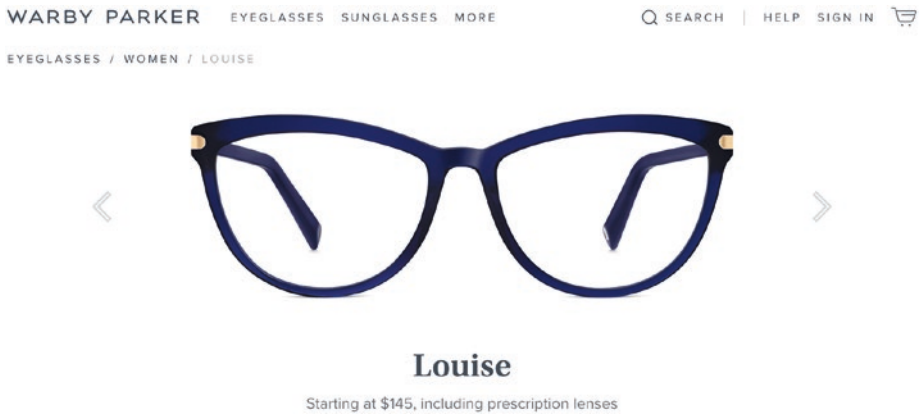


Figure 1-7. Screenshot of Warby Parker’s product images with no thumbnails

These images are the main subject of the page. There are fewer images to browse than most thumbnail collections (just three photos), so smaller previews would not save you much time. You can also predict what the next photo will be: the same glasses from another angle. You might as well jump straight to it than fill up the page with tiny thumbnails. This is an example of when you *don’t* need thumbnails.

Other pattern details

When describing a UI pattern, some people also specify these details:

- Why?
 - Explanation of how it solves the problem
 - Supporting principles, such as usability principles

- User research and other evidence
- Motivation
- Examples
- How it works
- Consequences
 - Trade-offs and drawbacks
 - Result context and expected improvements
- Implementation details
- Sample code or design assets
- Known uses “in the wild”
- Author or resident expert
- Related patterns
- Alternative names or aliases
- Links to more resources

These tend to be used to elaborate on the three main components—solution, problem, and context. Particularly—and importantly—it is common to suggest alternative, related patterns of interest in the context part of a pattern to clarify when *not* to use the pattern. I note these here so you may recognize them when you see them elsewhere and can consider them for yourself if you find yourself writing a pattern.

A specific collection of patterns for a project is often called a **pattern library**. Pattern libraries give teams a common language to improve their design processes. We’ll discuss pattern libraries further in [Chapter 4](#).

Why care about patterns?

UI patterns compare approaches, distilling the considerations and successes of designers before you. Knowing the patterns and understanding the decisions that went into them let you take advantage of the mounting wisdom of whole generations and industries that brought about these patterns, without reinventing the wheel. The small, reusable UI solutions found in these patterns can then be composed together to build cohesive, intuitive experiences that resonate with people.

Let's look at some of the other benefits of learning UI patterns.

Design efficiently

Knowing patterns can help you design efficiently by quickly recognizing the best tool for the job, understanding the value of different solutions, and solving the largest number of problems at once. For example, an autocomplete search box might help your site visitors navigate your site content, recognize the term they're looking for without knowing the exact name or spelling, and select a result after only typing a few characters without needing to waste energy typing in the full search term. By learning about the autocomplete UI pattern, you'll more quickly recognize when you need to use it, and likewise with all UI patterns. For example, if you need to redesign the navigation for a catalog of products by expanding the existing horizontal dropdown menu into a multilevel one, you can see how an autocomplete search box might solve the problem better. We'll look at autocomplete again in Chapter 3.

You can even recognize patterns across evolving technology. Compare the hated "Clippy" (www.theatlantic.com/technology/archive/2015/06/clippy-the-microsoft-office-assistant-is-the-patriarchys-fault/396653/) Microsoft Office assistant to Slack's chatbot called Slackbot, shown in Figure 1-8.

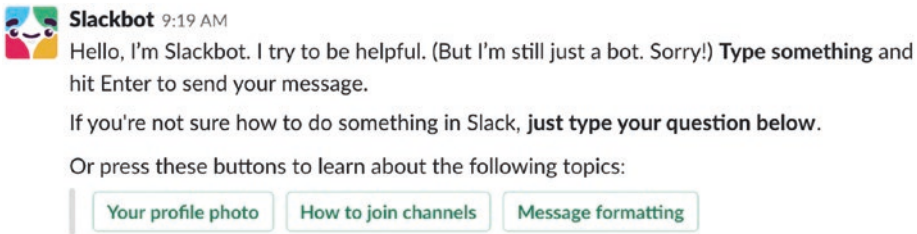


Figure 1-8. Screenshot of Slackbot introducing itself

As conversational UIs have developed, we've seen some drastic changes in how we interact with technology; however, we've also noticed some similarities. You can see some familiar patterns, including identity and profile information, private chat, and feed updates. Increasing your familiarity with diverse patterns will help you efficiently solve design problems you face in new user interfaces, especially if you understand the underlying usability principles and can adapt patterns to new contexts.

Consistency and familiarity

Using familiar patterns lets you foster predictability. The familiar idea of **drag and drop** lets you directly manipulate an object by dragging and dropping it. A common use of drag and drop is to upload an image by dragging it from your computer's local file system to a target drop area in the interface. Most to-do apps let you drag and drop to-do items to reorder them or move them to different lists. The more pervasive drag and drop interfaces become across the Web, the more likely people will understand how to interact with them. GitHub, for example, makes it clear that you can attach a file to a comment and that you can do this using several methods (drag and drop, select, paste). In Figure 1-9, you can see GitHub's rich text editor for comments that lets you drag and drop images.