University of Cincinnati

Date: 5/18/2012

I, Chelsea Chase, hereby submit this original work as part of the requirements for the degree of Master of Science in Computer Science.

It is entitled:
Using Design Patterns in User Interface Design

Student's name: Chelsea Chase

This work and its defense approved by:

Committee chair: Fred Annexstein, PhD

Committee member: Kenneth Berman, PhD

Committee member: Chia Han, PhD
Using Design Patterns in User Interface Design

A thesis submitted to the
Division of Graduate Studies and Research
of the University of Cincinnati

in partial fulfillment of the
requirements for the degree of

Master of Science

in the Department of Computer Science
of the College of Engineering
May 2012

by

Chelsea Chase

Thesis Advisor and Committee Chair: Dr. Fred Annexstein
Abstract

User interface development is a large part of software development and is highly visible to the user. It’s important to create a good user interface but it can be very time consuming. The use of design patterns can be beneficial to speed up user interface design, as well as implementation of common features. Design patterns are high-level solutions to common problems in software development. These patterns have been widely used in object-oriented programming and across many different programming languages. Each design pattern has a name and a set of attributes that make up the pattern. So far, no standard set of attributes has been agreed upon. We present an easy, all-inclusive list of attributes specifically developed for use with interaction patterns, the special type of design pattern used in user interface design. This list helps to bridge the gap left by other proposed attribute lists and we use it to define several design patterns identified in our case study of Facebook and its mobile app for iPhone and Android devices.
Acknowledgements

I would like to thank my advisor, Dr. Fred Annexstein, who has been invaluable throughout this process. I am grateful for Dr. Chia Han and Dr. Ken Berman serving on my thesis committee and to the Computer Science department at the University of Cincinnati for providing me with the opportunity to earn this degree through the ACCEND program. I would also like to thank Julie Muenchen, Eugene Rutz, and Dr. John Schlipf for helping me plan out my coursework and supporting my ambitious decisions. I give my sincere thanks to my friends and family for being there for me no matter how busy and stressed I was, especially to Bobby, for being my strength when I had none.
# Table of Contents

List of Figures ........................................................................................................ VI

1 Introduction ........................................................................................................... 1
   1.1 Importance of User Interface Development ................................................. 1
   1.2 Changing and Creating User Interfaces ....................................................... 1
   1.3 Introducing Design Patterns ....................................................................... 2
   1.4 Organization of this Thesis ......................................................................... 3

2 History of Design Patterns .................................................................................. 4

3 Layout of an Interaction Pattern ......................................................................... 7

4 Case Study: Facebook ......................................................................................... 9
   4.1 Interaction Patterns from the Facebook Website ......................................... 9
      4.1.1 Accordion Menu .................................................................................. 11
      4.1.2 Always-Visible Tools .......................................................................... 14
      4.1.3 Auto-Complete ................................................................................... 17
      4.1.4 Availability Indicator .......................................................................... 21
      4.1.5 Breadcrumbs ....................................................................................... 24
      4.1.6 Calendar Picker .................................................................................. 27
      4.1.7 Continuous Scrolling .......................................................................... 29
      4.1.8 Hover-Reveal Tools ............................................................................ 33
      4.1.9 Input Prompt ....................................................................................... 37
      4.1.10 Overlay .............................................................................................. 40
      4.1.11 Reflector ............................................................................................ 43
      4.1.12 Sign-In Continuity ............................................................................. 46
      4.1.13 Tag ..................................................................................................... 49
      4.1.14 Terms of Service Button .................................................................... 53
      4.1.15 User Card .......................................................................................... 56
   4.2 Interaction Patterns from the Facebook Mobile Apps .................................... 59
      4.2.1 Carousel .............................................................................................. 60
      4.2.2 Expandable List ................................................................................... 64
      4.2.3 Fling to Scroll ....................................................................................... 67
      4.2.4 Mode Selector ..................................................................................... 70
List of Figures

Figure 4.1-1: An accordion menu in Microsoft Outlook’s web application. ........................................... 11
Figure 4.1-2: A minimalistic accordion menu from RecycleNow.com. ...................................................... 12
Figure 4.1-3: Picasa’s use of an accordion menu with indicator arrows. .................................................. 12
Figure 4.1-4: Facebook’s comments view is similar to an accordion menu. .............................................. 13
Figure 4.1-5: Facebook’s Movies view before being expanded ................................................................. 13
Figure 4.1-6: Facebook uses an accordion list to display more photo albums ........................................ 14
Figure 4.1-7: YouTube includes always-visible tools above and below the video ..................................... 14
Figure 4.1-8: Gmail always shows star (favorite) and tag (important) tools next to each email ............... 15
Figure 4.1-9: Gmail’s mobile app also has the always-visible star tool but only a tag indicator ............... 16
Figure 4.1-10: A Facebook link post with always-visible Like, Comment, and Share action links ........... 16
Figure 4.1-11: Facebook Mobile’s wall post with standard action links .................................................. 17
Figure 4.1-12: Google’s auto-complete is based on popularity of the search terms ............................. 17
Figure 4.1-13: Apple’s website uses a product and site search with auto-complete ............................... 19
Figure 4.1-14: Yahoo! Finance uses Auto-complete to help suggest stocks ........................................... 19
Figure 4.1-15: Two examples of Facebook’s auto-complete feature ....................................................... 20
Figure 4.1-16: Gtalk’s mobile app displays an availability indicator with multiple icons .................. 21
Figure 4.1-17: GMail’s Gtalk module shows a set of availability icons similar to the mobile app ........ 23
Figure 4.1-18: The entry on chat availability in Facebook’s Help area .................................................. 23
Figure 4.1-19 (left): Facebook’s chat sidebar with friends online, on mobile apps, and offline ........... 24
Figure 4.1-20: Facebook’s mobile Messenger app uses the same icons as the web application ........... 24
Figure 4.1-21: The Windows 7 control panel uses breadcrumbs to aide in navigation .......................... 24
Figure 4.1-22: Amazon’s breadcrumbs list product categories .............................................................. 25
Figure 4.1-23: EBay’s breadcrumb navigation for auction/shop categories ............................................ 25
Figure 4.1-24: The breadcrumbs feature on a Facebook Timeline ....................................................... 26
Figure 4.1-25: Microsoft Outlook’s calendar picker for scheduling appointments ............................. 27
Figure 4.1-26: Travelocity’s Android calendar picker ............................................................................. 28
Figure 4.1-27: Travelocity’s web calendar picker shows the entire date range ..................................... 28
Figure 4.1-28: The calendar picker for setting Facebook event dates .................................................... 29
Figure 4.1-29: Humanized.com displays a message when more posts are being loaded ...................... 30
Figure 4.1-30: Twitter displays an animated “loading” circle while loading more tweets ................... 31
Figure 4.1-31: Pinterest displays a bouncing ball with the label “Fetching pins…” ................................ 31
Figure 4.1-32: Facebook’s news feed loading animation ................................................................. 32
Figure 4.1-33: Facebook’s Timeline loading and right-side date indicators ............................................. 33
Figure 4.1-34: Twitter’s tools for manipulating individual tweets are hover-reveal tools ................... 34
Figure 4.1-35: Google’s use of hover tools for search results include +1 and preview (>>) buttons ...... 35
Figure 4.1-36: Facebook’s hover tools for a user’s own Timeline ........................................................... 36
Figure 4.1-37: Hover tools used on a user’s Facebook Timeline profile picture ................................... 36
Figure 4.1-38: Hovering over an item on the Facebook news feed reveals a dropdown tool. ......................................................... 37
Figure 4.1-39: Twitter uses input prompts for general and specific searches. .......................................................... 37
Figure 4.1-40: Twitter’s login screen provides prompts for every field. .......................................................... 38
Figure 4.1-41: Facebook’s search and status input prompts. .................................................................. 39
Figure 4.1-42: Facebook’s comment input prompt. .................................................................................. 39
Figure 4.1-43: Groupon displays an overlay message to welcome new users. .................................................. 40
Figure 4.1-44: Flicker allows users to view a photo’s location on an overlay map. ......................... 41
Figure 4.1-45: Facebook’s event overlay returns the user to the community events page. .......... 42
Figure 4.1-46: An example of a user card reflector by Yahoo! ................................................................. 43
Figure 4.1-47: Facebook’s reflector pane allows viewing by specific friends or the general public. 44
Figure 4.1-48: The View As... button puts the Timeline into a reflector mode. ................................................. 45
Figure 4.1-49: A Timeline in reflector mode shows the user how it appears to the general public. 45
Figure 4.1-50: YouTube allows sign-in continuity so that users can verify their age. ......................... 46
Figure 4.1-51: Kayak opens a login overlay (see section 4.1.10) to implement sign-in continuity. .......... 47
Figure 4.1-52: Visiting the Timeline for TV show Futurama prompts the user to sign in. .................... 48
Figure 4.1-53: Vimeo.com allows users to add and remove video tags. ............................................................. 49
Figure 4.1-54: Twitter lists trending topics and hash tags by local area. ................................................. 50
Figure 4.1-55: Adding a tag to a photo. ..................................................................................................... 51
Figure 4.1-56: Hovering over a tag to display the tagged region. .............................................................. 51
Figure 4.1-57 (Left): Hovering over the photo region to display the tag. .................................................. 52
Figure 4.1-58 (Above): Adding a tag using the @ symbol. ................................................................. 52
Figure 4.1-59: Sorting all Facebook album photos based on one user’s tag. .............................................. 52
Figure 4.1-60: An example of a Terms of Service button by Yahoo! ......................................................... 53
Figure 4.1-61: A standard Mac installer prompts the user to agree to the terms of service. ................... 54
Figure 4.1-62: Facebook’s sign up page uses a terms of service button indicated in the fine print. 55
Figure 4.1-63: LinkedIn’s version of a user card. ....................................................................................... 56
Figure 4.1-64: Digg’s user cards for a source. ............................................................................................ 57
Figure 4.1-65: Gmail’s chat user card. ....................................................................................................... 57
Figure 4.1-66: Facebook’s user card displayed by hovering over a user’s status update. ....................... 58
Figure 4.1-67: The user card of a friend in a special friend group. ............................................................. 58
Figure 4.2-1: The Android carousel pattern from AndroidPatterns.com ................................................. 60
Figure 4.2-2: Amazon’s mobile suggestions are listed in a carousel. ....................................................... 61
Figure 4.2-3: Between the first two photos in the carousel. ................................................................. 63
Figure 4.2-4: Focused on the second photo in the carousel. ................................................................. 63
Figure 4.2-5: After moving past the second photo in the carousel. ........................................................ 63
Figure 4.2-6: The Android expandable list pattern from AndroidPatterns.com ..................................... 64
Figure 4.2-7: An android browser’s history before being expanded. ..................................................... 65
Figure 4.2-8: An Android browser’s history after being expanded. ........................................................ 65
Figure 4.2-9: The bottom of the current month on a Facebook Timeline, before being expanded. .......... 66
Figure 4.2-10: The Timeline after April has been expanded. ................................................................. 66
Figure 4.2-11: The Android fling to scroll pattern from AndroidPatterns.com ...................................... 67
Figure 4.2-12: All of Twitter’s mobile pages implement fling to scroll. .................................................. 69
Figure 4.2-13: Google+’s profile page is fling to scroll .................................................................69
Figure 4.2-14: Landscape scrolling leaves less room for the user to gesture on the screen............70
Figure 4.2-15: Portrait mode is the typical mode for fling to scroll up/down a list ....................70
Figure 4.2-16: The Android mode selector pattern from AndroidPatterns.com..........................71
Figure 4.2-17: Twitter’s filters update which mode the list is in ..................................................72
Figure 4.2-18: Engadget moves the screen down when the mode indicator is selected ...............72
Figure 4.2-19: Selecting the newspaper icon to bring up the dropdown menu .........................74
Figure 4.2-20: The dropdown menu overlay for selecting a mode for the news feed .................74
Figure 4.2-21: The Android pull to refresh pattern from AndroidPatterns.com .......................75
Figure 4.2-22: Twitter’s first pull to refresh message .................................................................76
Figure 4.2-23: Twitter’s second message for the user to release to refresh ................................76
Figure 4.2-24: Twitter’s third message indicates success ............................................................77
Figure 4.2-25 (left): After the user scrolls past their cover photo ..............................................77
Figure 4.2-26 (right): Once the user has pulled the instruction down .........................................77
Figure 4.2-27: Successful refresh of the page displays a spinning indicator during the refresh....78
Chapter 1  Introduction

Section 1.1  Importance of User Interface Development

A user interface, or UI, is the way in which a user can use and interact with a software program. It's obvious that some kind of UI is necessary for every piece of software but for most programs, a graphical user interface, or GUI, is used. A GUI allows users to see graphical representations of program options and is what we are all used to using today. The traditional representation of a software program to the average user always has a graphical user interface. Users expect easy-to-use controls with descriptive labels and clear formatting. Once a user has learned how to use the GUI of one program, they often relate back to it when facing a new program which has similar controls. Programs which are too complicated or too cluttered are often regarded as bad programs, even if the functionality is has no issues. Therefore, it's important to develop a user interface which caters to the learning curve of a user as well as the functionality it provides. A well-developed user interface is essential to creating a successful software application.

Section 1.2 Changing and Creating User Interfaces

Just as users prefer new programs to be similar to the old ones they have used (to help them learn the new controls), they also prefer new versions of programs to keep similar GUI layouts. Applications that change too often will lose users and/or gain negative publicity. One such example is the ongoing changes that the popular social network, Facebook, has made over the years. Drastic changes have been met with overwhelming opposition and some users have gone as far as to stop using Facebook completely because a new much different user interface is forced on them. Facebook is notorious for this because of the frequency of the changes they make, adding a new feature or recreating an old one before users can even adjust to the current features. The problem is not that the user interface is changed to something less user-friendly, it's only the frequency and magnitude of the changes that users are typically upset about. Users do not want
to have to relearn software applications often, or in large chunks. The problem for user interface designers is that when a GUI is outdated, it’s difficult to make changes to update it without upsetting the userbase.

Another problem for those who face designing a GUI for a software application is the way in which the GUI is built. Many programming languages have libraries dedicated to GUI development. However, these libraries only contain parts and pieces of a whole GUI and it is up to the developer to choose how to put these pieces together. Often, a software architect or a user interface designer decides what the GUI will look like before the developer starts coding; in these cases, the developer must try to match the intended GUI as best he/she can. Other times, independent developers will come up with their own GUI layouts and develop them themselves.

In either situation, someone must figure out what tools the user will want to use in order to interact with the program, and how each of those tools are laid out on the screen. One way to do this is to see what similar programs have looked like and then develop the new program in a like manner. The advantage to doing it this way is that users often see the same layouts for different programs and can easily adapt to using the new program. Unfortunately, researching similar programs takes time and only results in the design of the GUI, not the underlying code.

Section 1.3  Introducing Design Patterns

Jennifer Tidwell stated that “a pattern describes possible good solutions to a common design problem within a certain context, by describing the invariant qualities of all those solutions” when she first began studying design patterns at MIT [T99]. Since then, she has gone on to publish books on designing interfaces and has written many design patterns herself. This basic explanation of a pattern helps us to derive that a design pattern is a construct created to solve a particular problem or set of problems that are commonly occurring in a specific field. In computer
science, design patterns are often linked with object-oriented design and are essentially collections of knowledge on object-oriented methods. Design patterns are also beginning to be useful in the design of user interfaces. Instead of a designer looking at similar sites and researching how popular applications implement certain design constructs, they can consult a design pattern library for solutions to common problems, examples, and sometimes pseudocode.

The father of the design pattern, Christopher Alexander defined his original design pattern concept as “a three-part rule, which expresses a relation between certain context, a problem, and a solution.” He goes on to say “Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem.” Alexander used these beliefs as first step in writing his own design patterns for architecture. We will later present a design pattern schema that is based off of the attributes used in Alexander’s and others’ design patterns.

**Section 1.4 Organization of this Thesis**

In this thesis, we will discuss the history of the design pattern in greater detail from the beginning to present day studies. We have developed a list of attributes we believe should be included in each interaction pattern and will present these in chapter 3. We will then apply the current knowledge of design patterns, and specifically interaction patterns, to our case study in chapter 4. We study the use of interaction patterns by the popular social networking application, Facebook. We will identify over 20 interaction patterns that Facebook uses and then write out each pattern according to our design pattern schema. Lastly, we will briefly discuss conclusions and future applications of this work.
Chapter 2 History of Design Patterns

By far the most important name in the history of design patterns is Christopher Alexander. He is considered the “father of the Pattern Language movement in computer science” due to his work in design patterns for architecture. Alexander is a world-renowned architect and has published many works on a variety of architectural subjects but his book “A Pattern Language” is what sparked interest in the Computer Science field [P1]. The book was published in 1977 and contained a general description of a pattern for use in architectural design:

"Each pattern describes a problem which occurs over and over again in our environment, and then describes the core of the solution to that problem, in such a way that you can use this solution a million times over, without ever doing it the same way twice." [A77]

It was this concept that inspired the paper “Using Pattern Languages for Object-Oriented Programs”, co-published by programmer Ward Cunningham of Tektronix and Kent Beck of Apple. The paper was written for the 1987 Object-Oriented Programming, Systems, Languages & Applications (OOPSLA) conference organized by ACM [BC87]. The idea gained popularity and soon, design patterns in object-oriented programming became a growing trend. Cunningham created the “Portland Pattern Repository” as a collection of design patterns he and others had come up with. Later on, he decided that to help other developers share their patterns, he would host a website where they could submit them for feedback- interestingly, this was the creation of the very first wiki called Wiki Wiki Web [C1][C2]. Cunningham began programming the wiki in 1994, a big year for design patterns in software engineering.

It was around this time that design patterns gained enough popularity to merit a conference of their own called the Pattern Languages of Programs (PLoP). The PLoP conferences have been sponsored annually by the Hillside Group and ACM since their inception in 1994 [H1]. Later in the
same year, the most influential book on design patterns (still to this day) was published by the Gang of Four, a group of four authors: Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. The book was called “Design Patterns: Elements of Reusable Object-Oriented Software” and has since been shortened to design patterns book, book by the gang of four, and GOF book [C3]. This book has been cited by over a hundred other publications and although a second edition was never written, the first edition is still popular enough to have been digitalized recently and made available as an e-book for the Kindle e-reader, as well as various other digital formats [A4].

The book by the Gang of Four did not include any design patterns for use in user interface development [G95]. UI development was a logical next step for the concept of design patterns to be used in after their usage was made popular in more behind-the-scenes programming. The self-proclaimed “first substantial set of user-interface patterns” was the collection created by Jennifer Tidwell called “Common Ground” [D1]. Tidwell proposed the use of design patterns in HCI design while she was at MIT and submitted her resulting pattern language to PloP ‘98. "Common Ground" categorized design patterns by questions such as “How can the user navigate through the artifact?” and sometimes named patterns in generally unhelpful ways. The library was last updated in 1999 and was never completed (not all the patterns have required content) [T99]. A new resource was introduced when Tidwell wrote a book on the subject called “Designing Interfaces” in 2005. The new website categorizes the patterns in a better way and includes screenshots for many of the patterns listed. Because the field is always changing and being updated, a second edition was published only five years after the first, in 2010. This included new design patterns and also left out old ones that are so standard now, no other method is in use to solve the same problem. Today, this book is the most accurate representation of interaction patterns. Unlike the first edition, the accompanying website does not have a catalog of the design
patterns from the book, but only presents a select few [T1]. Many of Tidwell’s design patterns can only be seen by purchasing her book, which makes it more difficult for those patterns to be as commonly used as more publicly available ones.

This brings us to the current pattern libraries available to the public today. There are four main libraries online which we will reference in each design pattern. They are: The Yahoo! Design Pattern Library [Y1], UI-Patterns.com [T1], PatternRy [P2], and Quince [Q1]. There is also a library devoted to Android design patterns which we will reference in our mobile design patterns, it’s called AndroidPatterns.com [A3]. Each of these online libraries contains different patterns, some overlapping, and their own renditions of how a pattern should be written. They are all recently updated and have screenshots of currently used applications, some of which are used in this thesis. The newest edition of Designing Interfaces does not have an accompanying pattern library like the old edition did. Instead, only a few select patterns were published online. These are the most current sources on interaction patterns today, we’ve used much of this research to conduct and interpret our own, hopefully in the future the libraries will converge into one resource to aid UI designers everywhere.
Chapter 3  Layout of an Interaction Pattern

Patterns specifically referring to HCI can be called HCI patterns, UI patterns or most traditionally, interaction patterns. They are classified by category like all design patterns and share the same attributes of name, problem, solution, and consequences. Effective design patterns have been found to need more than just those four pieces though. For interaction design patterns, an example screenshot is required in order for a pattern to be easily understood. Patterns should also list the context and rationale for the use of the specific design.

Because so many different ‘standards’ exist for the description of interaction patterns, it’s important to clarify the way in which we will explain each pattern. A simple scheme, adapted from a combination of Jennifer Tidwell's *Designing Interfaces* and the Gang of Four’s *Design Patterns* is one that is familiar to many people already, just not used in an interaction pattern—who, what, when, where, why, and how. These attributes each describe an essential part of an interaction pattern as described below:

- Who it’s for: The categories the pattern falls into, based on the types of programs that should use this design, such as social networks or business applications.
- What it is: The general description of the pattern, potentially with a screenshot and explanation of the features.
- When it should be used: The context the pattern should be used in, specifically, which problems this pattern solves.
- Where it can be used: The platform the pattern is ideal for, desktop, web, mobile, etc.
- Why it works: The way the pattern solves the problem at hand.
- How it’s used by others: Screenshots and examples of popular applications that have used this pattern in the past, as well as libraries/collections that this pattern can be found in.

Combined, they form a complete interaction pattern which adequately describes all aspects of the pattern and provides ample room for additions and examples. Now that design patterns, and
interaction patterns in general, have been adequately described and their histories have been laid out, let's apply this knowledge and the newly proposed pattern attributes to a case study.
Chapter 4  Case Study: Facebook

Section 4.1  Interaction Patterns from the Facebook Website

Facebook is a popular social networking website in which users can connect with others in their local area, their school, their workplace or any other group. Once a user has registered, they can create their own Facebook Timeline, as well as view the content of other users’ Timelines, like photos and status updates. Users can modify the privacy of any single post, or set defaults for all general areas, such as photo albums, wall posts, tagged photos/videos, etc. This allows users to control exactly who sees what information, protecting the user's privacy but still allowing them to share information with others. Although privacy is a very important topic in all social networks, there are other key features of Facebook that help it to be the overall largest social network. These features include a live chat mechanism, the ability for third party developers to create apps for use on Facebook, and a number of user friendly interface features, which is what we are most concerned with in this thesis.

Facebook was founded in 2004 as a website for college students only [F12]. Because of its popularity, Facebook has now expanded to include users of all ages and backgrounds. Although it was originally launched in English, there are now more than 70 different languages Facebook can be used in, and people from many countries have Facebook accounts [F2]. The growth of Facebook has been astounding and it is now the second most visited website in the world [A1]. Facebook’s mobile application is one of the top free apps for iPhones and iPads and for Android devices. It is for these reasons Facebook was chosen for our case study. In this study, we analyze Facebook’s user interface and identify the interaction patterns used. We then present each interaction pattern, in full, followed by an explanation of how Facebook specifically uses the pattern to better the user experience.
The following patterns will be presented in this section:

4.1.1 – Accordion Menu

4.1.2 – Always-Visible Tools

4.1.3 – Auto-Complete

4.1.4 – Availability Indicator

4.1.5 – Breadcrumbs

4.1.6 – Calendar Picker

4.1.7 – Continuous Scrolling

4.1.8 – Hover-Reveal Tools

4.1.9 – Input Prompt

4.1.10 – Overlay

4.1.11 – Reflector

4.1.12 – Sign-In Continuity

4.1.13 – Tag

4.1.14 – Terms of Service Button

4.1.15 – User Card
4.1.1 Accordion Menu  
A.K.A. Inline Expand, Expandable List (mobile)

Who it’s for
All types of applications can use this pattern.

What it is
A menu which can collapse to show less data, or expand to show more at the click of the mouse.

![An accordion menu in Microsoft Outlook's web application.](image)

Figure 4.1-1: An accordion menu in Microsoft Outlook’s web application.

When it’s used
When screen real estate is highly sought after, displaying large amounts of data can be costly. Applications which deal with large amounts of often segmented data can have trouble displaying it without causing the application to look cluttered. Windows that have too much information in them often cause users to turn away, especially if much of the information is not necessarily something the user is interested in. Often, data can be shrunken down to show only a highlight if the user is not interested in the details but an application has no way of knowing what a user would like to see detailed information on, and what they would like to see only highlights of.

Where it’s used
Accordion menus can be found on the web, in desktop applications, and most commonly on mobile devices.

Why it works
Accordion menus can take a large amount of data and ‘hide’ it in the accordion. Then when users initially look at the menu item, all they see are item’s highlights- a short description and the indication that this menu may be expanded to show more details. If the user wishes to expand the
accordion, they simply click the button to do so (normally a +) or in the case of mobile devices, tap the menu item itself. Expanding the accordion should do so in-frame, without needing to refresh the page. The user should then be able to shrink the menu again by clicking the same button (this time showing a -) or tapping the header of the menu item. This process allows users to choose what information is on a given page, but also allows the potential to display more information and maximize screen real estate. This is especially valuable in mobile devices because screens are so much smaller.

**How others have used it**

Other popular applications that use accordion menus include Gmail, Picasa, Yahoo!, Groupon Mobile, Facebook, and Facebook Mobile.

![Figure 4.1-2: A minimalistic accordion menu from RecycleNow.com.](http://ui-patterns.com/patterns/AccordionMenu)

![Figure 4.1-3: Picasa’s use of an accordion menu with indicator arrows.](http://patternry.com/p=inline-expand/)

The Accordion Menu can be found in these libraries/collections:

The Accordion Menu as used in Facebook

Facebook uses the accordion menu in multiple places, most notably in the comments portion of each post (Figure 4.1-4) where the user can choose to view all comments to expand the comment list. Once clicked, Facebook's accordion menus cannot be re-collapsed, which is different from a traditional implementation. This is similar in implementation to Expandable Lists, a mobile design pattern seen later in section 4.2.2. Other areas that behave like this are viewing the Likes page (Figure 4.1-5 shows Movies likes) and viewing a collection of Photo Albums (Figure 4.1-6).

Figure 4.1-4: Facebook’s comments view is similar to an accordion menu.

Figure 4.1-5: Facebook’s Movies view before being expanded.
4.1.2 Always-Visible Tools

A.K.A. Always-Visible Controls, Visible Tools

Who it’s for
Any type of application can use this pattern.

What it is
A set of buttons, links, or otherwise selectable utilities which cannot be hidden and are available under every circumstance.

Figure 4.1-6: Facebook uses an accordion list to display more photo albums.

Figure 4.1-7: YouTube includes always-visible tools above and below the video.
When it’s used
When a user is browsing an application and wants to interact with an object, there are a variety of ways they might do so. Each application implements tools that users can use in a different way but oftentimes it is easiest for the user if they can see what their options are right away. If a tool is hidden in a dropdown menu or a context menu, the user may not know that it is available. Likewise, if a set of tools appears only when the user hovers over the object they want to interact with (see section 4.1.9) it is possible that the user will miss those tools altogether, if they don’t know to look for them.

Where it’s used
Always-visible tools are useful on all platforms, especially on mobile devices which cannot use hover-reveal tools.

Why it works
If a user can view an object and instantly see the actions that are available to perform on it, it removes the need for the user to explicitly learn how to interact with that object. Removing the need for walk-throughs and tutorials makes learning an application easier and faster. Always-visible tools are intuitive and require no action on the user’s part in order to display. Everything is laid out in one place, in plain sight, and ready for interaction with the user.

How others have used it
Other popular applications that use always-visible tools include Gmail, Facebook, YouTube, Amazon, and their respective mobile apps.

Figure 4.1-8: Gmail always shows star (favorite) and tag (important) tools next to each email.
Always-visible tools can be found in these libraries/collections:

http://patternry.com/p=always-visible-tools/

**Always-visible tools as used in Facebook**

Facebook implements always-visible tools most notably on user posts found on a user's news feed or Timeline. These posts can have a variety of content, including status updates, photos albums, new friend additions and information changes (relationships, life events, etc.), even profile picture changes are posted to users’ news feeds. Every type of post will always contain both comment and like action links. Some posts can be shared, like wall photos, or links; in these cases, there is also a share link available as seen in figure 4.1-10. These tools are never hidden and appear at the bottom of every post whether the user is viewing it on the news feed, Timeline, or the individual. Facebook mobile also provides these tools as seen in figure 4.1-11.

**Figure 4.1-10:** A Facebook link post with always-visible Like, Comment, and Share action links.
4.1.3 Auto-Complete

A.K.A. Autocompletion, Search Suggestions, Continuous Filter, Entry Suggestions, Inline Suggestions, Type Ahead

Who it’s for
Applications with user input forms and/or search functionality can use this pattern.

What it is
Auto-Complete is the mechanism that matches a user’s query with potential matches and suggests terms as the user types into a field. This is commonly found in forms and search fields.

Figure 4.1-11: Facebook Mobile’s wall post with standard action links.

Figure 4.1-12: Google’s auto-complete is based on popularity of the search terms.

Screenshot from UI-Patterns.com
When it’s used
Entering information can be tedious for users, especially if they are trying to enter a long or complicated item into a field. It’s frustrating to make typos and have to re-type queries, especially when the input is a common one. Email addresses, names, URLs, and home/business addresses are all very common inputs, but are often annoying to have to spell out. Sometimes, a user may not even know what exactly it is they are trying to enter, like when searching, so it’s difficult to get the right results without wasting time and effort.

Where it’s used
Auto-complete is used mostly on the web and game consoles, but can be used on other platforms as well.

Why it works
Auto-Complete fills in data as the user types, allowing them to type out only part of their entry and select the correct one from a list of suggestions. This format bypasses many typos (and can sometimes correct them) and suggests popular results so that the user can choose what’s best for them. As the user inputs more information, the suggestions made by auto-complete get narrower, helping to indicate when the user has found the best term for their search or other query, even if they couldn’t remember the exact phrase off the top of their head. By having this mechanism do a large part of the work, the user can move faster and with fewer errors, making them more efficient and less frustrated.

How others have used it
Other popular applications that use auto-complete include Google, Yahoo!, Amazon, Facebook and Microsoft Outlook.
Figure 4.1-13: Apple’s website uses a product and site search with auto-complete.

Screenshot from UI-Patterns.com

Figure 4.1-14: Yahoo! Finance uses Auto-complete to help suggest stocks.

Screenshot from UI-Patterns.com

Auto-Complete can be found in these libraries/collections:

http://ui-patterns.com/patterns/Autocomplete (screenshots source)
http://patternry.com/p=autocomplete/
http://quince.infragistics.com/#/Search/ViewPattern$pattern=Text+Field+Autocompletion
Auto-complete as used in Facebook

Facebook uses auto-complete in their search field. A user can type in a portion of a name or term and Facebook will bring up apps, people, pages, groups, and Facebook links that may be related to the term so far. This is especially useful for names and locations since it doesn’t require perfect spelling or remembering the entire name of the person/place, like in figure 4.1-15. Facebook also employs user information in their auto-complete suggestions; for instance, when a user who has listed the University of Cincinnati as his school types in “University of” into the search bar, the University of Cincinnati makes up the top search results because it’s more likely that that particular user is looking for their own school’s related pages or groups. If the user (or their friends) don’t have specific information that is similar to their query, relative location is used so that users from Ohio who are searching for a person will see other people close to that name that are nearby before seeing someone by that name in a different state or country.

![Figure 4.1-15: Two examples of Facebook's auto-complete feature.](image-url)
4.1.4 Availability Indicator

Who it’s for
Applications which include direct communication (message, call, email, etc.) can use this pattern.

What it is
An indicator next to a user’s name or email address which shows the user whether that user is currently available.

![Figure 4.1-16: Gtalk’s mobile app displays an availability indicator with multiple icons.](image)

When it’s used
If a user wants to get in touch with another specific user, it’s often important for them to know if that user is available at the moment they wish to send them a message (or speak with them). It can be troubling when an important message needs to be passed along and the user does not know if their intended recipient is going to receive it right away. Users prefer to know which of
their contacts, or other users of the application, are online and which are unreachable at the moment. In the case of instant messaging/chat programs, this is necessary, but not all users who are online are currently available or ready to communicate. In these cases, a user may be required to guess at whether their recipient is actually able to respond right away or not.

**Where it’s used**
The availability indicator can be found on web and desktop applications, mobile devices, and game consoles.

**Why it works**
A simple color-coded indicator can flag users as online/available, idle/busy, or offline/unavailable. This three-stage availability indicator is the minimum needed for a user to know whether now is a good time to send another user a message. For phone calls, this can reduces the number of calls that lead to a voicemail or failed call. For text messages, this indicates the likelihood of an immediate reply. Other indicators can be used, such as a status message or even an event calendar for each user (indicating when they are busy and when they are intending to be available). Users can weight their decision to send a message based on the recipient’s indicator, which takes much of the guesswork out of sending without it.

**How others have used it**
Other popular applications that use availability indicators include the web and mobile versions of Gtalk, Yahoo! Mail, and Microsoft Outlook, the desktop and mobile versions of Skype, and on game consoles such as the Sony Playstation 3.
The Availability Indicator can be found in these libraries/collections:


The availability indicator as used in Facebook
The right-hand side of Facebook normally contains a split pane which features a chat interface in the bottom half. This lists a user’s friends which have been frequently or recently interacted with no matter what their current chat status is. Users who are on Facebook at the moment have a green dot next to their name, users who have downloaded the Facebook Mobile app will have a mobile phone icon, and offline users will have no icon (figure 4.1-18). Underneath this list is an additional list of online friends which do not interact with the user enough to be in the top list. To find a friend quickly, the user can type that person’s name into the search box at the bottom of the list (figures 4.1-19 and 4.1-20). Offline users can still receive messages later, which is why they are displayed in this list (if the user interacts with them frequently enough).

Figure 4.1-18: The entry on chat availability in Facebook’s Help area.

Figure 4.1-17: Gmail's Gtalk module shows a set of availability icons similar to the mobile app.

Use the symbols next to your friends' names to figure out who's available to chat:

- Friends with a • next to their names are available to chat
- Friends with a □ next to their name have downloaded one of the Facebook apps (ex: Messenger or Facebook for iPhone)
- Friends with no icon next to their names are unavailable
4.1.5 Breadcrumbs  
A.K.A. Breadcrumb Trail, Train Marker, Homeward Path

Who it’s for
Any type of application can use this pattern.

What it is
A series of labels that indicate the user’s location in the context of the application.

Figure 4.1-21: The Windows 7 control panel uses breadcrumbs to aide in navigation.
When it’s used
When users are navigating a website or large application, it’s often difficult for them to remember where in the context of the application they have navigated to. This can cause problems even with the ability to use a browser’s back button (in the case of web applications) because the user cannot see the hierarchy of the application, only the order in which they visited the pages. A user should also have the option to back up to a desired area of the application and see easily where they puts them in relation to their current page.

Where it’s used
Breadcrumbs are normally used on the web and in certain desktop applications (mainly operating systems); it is possible for breadcrumbs to be on mobile devices but it’s not typically used.

Why it works
Using the breadcrumbs pattern to help facilitate easy navigation allows the user to see at a glance where their current location is in the hierarchy of the application. The pages are simply named and displayed in an intuitive format (normally left to right for the United States) and previously visited pages up the hierarchy are just a click away. Users can navigate back to a more general place to choose a different path, or to complete a different activity from a central hub.

How others have used it
Other popular applications that use breadcrumbs include Windows 7, Facebook, Yahoo!, Amazon, and EBay.

Figure 4.1-22: Amazon’s breadcrumbs list product categories.

Figure 4.1-23: EBay’s breadcrumb navigation for auction/shop categories.
Breadcrumbs can be found in these libraries/collections:

http://ui-patterns.com/patterns/Breadcrumbs
http://patternry.com/p=breadcrumbs/

**Breadcrumbs as used in Facebook**

Facebook is typically only centralized around one main page (the news feed) and several subpages such as users’ Timeline (profile) and photo albums. Because of this, the use of the breadcrumbs pattern is adapted for navigating between times in a Timeline rather than between pages. A user’s timeline contains all the content that’s related to them since the time that they joined Facebook, as well as some retroactive content such as a user’s birth event. When a user scrolls down a timeline, their own or another user’s, they can view highlights of past months or years or view all the activity for that particular time. This is where breadcrumbs come into play, Facebook includes a breadcrumbs feature as seen in figure 4.1-24 that activates when a user scrolls down past the Timeline’s header and shows where the user is in relation to the entire timeline. The user can then use the feature to jump back to the current time, or to jump to a different time, forward or backward. The ability to jump forward is another aspect that differs from the traditional breadcrumbs implementation, but is a very useful addition in this case.

![Figure 4.1-24: The breadcrumbs feature on a Facebook Timeline.](image)
4.1.6 Calendar Picker A.K.A. Date Picker

Who it’s for
Any application with a date field can use this pattern.

What it is
A dropdown monthly calendar which the user can select a date from in order to populate a date field.

![Calendar Picker Example](image)

*Figure 4.1-25: Microsoft Outlook’s calendar picker for scheduling appointments.*

Screenshot from Quince.infragistics.com

When it’s used
Not only are there multiple formats to write a date in, which differ across the world, but there are also different delimiters that a date field might expect. For instance, the date July 2\(^{nd}\), 2012 may be written as 07-02-2012 or as 7/2/12 in the United States, or in European countries it could be 02-07-2012 or 2/7/12. Oftentimes, people don’t consider the format a field is looking for when typing dates and this can lead to bad information (such as reading 2/7/12 as February 7\(^{th}\)) and frustration for the user. Adding hints in the field or nearby such as mm/dd/yyyy is helpful but not always noticed by the user.

Where it’s used
The calendar picker is equally beneficial on all platforms.
Why it works
The format of a calendar is universal- all countries use the same calendar, despite their date formats. By providing a calendar as a dropdown, it gives users the choice to select their date from the calendar by navigating to the month and selecting the correct day. This is especially useful for dates that are near today’s date, since that is normally the starting month of the calendar picker, with today’s date highlighted for convenience. The calendar picker can then insert the selected date into the date field, correctly formatted with no worry about misrepresented data. Because the calendar picker only inserts the date into the field, it can normally be modified afterward in case the customer would like to change the year without navigating to that year in the picker itself (this is helpful for birthdate fields). Overall, the convenience allows customers to accurately enter data without spending too much time selecting from the calendar picker.

How others have used it
Other popular applications that use calendar pickers include Google, Amazon, Twitter, Facebook, Travelocity and their respective mobile applications.

Figure 4.1-26: Travelocity’s Android calendar picker.

Figure 4.1-27: Travelocity’s web calendar picker shows the entire date range.
The Calendar Picker can be found in these libraries/collections:

http://ui-patterns.com/patterns/CalendarPicker
http://quince.infragistics.com/#/Search/ViewPattern?pattern=Date+Picker (figure 4.1-25 source)

The calendar picker as used in Facebook
Facebook uses the calendar picker in a traditional way for entering the date of an event. The event creation page shows today’s date and a small calendar symbol in the date field; when the field is clicked, the calendar picker automatically shows with the specified date indicated by a gray box as seen in figure 4.1-28. The user can select a date from the picker, or type one into the field. As the user types, the calendar updates to show the correct month and highlight the correct day.

Figure 4.1-28: The calendar picker for setting Facebook event dates.

4.1.7 Continuous Scrolling
A.K.A. Endless Scrolling, Dynamic loading of a List (mobile)

Who it’s for
Any type of application can use this pattern, especially those with large amounts of data.
**What it is**

An alternative to pagination, continuous scrolling is the ability to scroll down and dynamically load information without having to stop or go to the next page of data.

![Figure 4.1-29: Humanized.com displays a message when more posts are being loaded.](image)

**When it's used**

For applications with large amounts of data, it is often appropriate to load that data in chunks. This is especially useful for data which is typically sorted by relevancy (most relevant pieces of data are shown first). Users can then move to the next chunk of loaded data if they haven’t found what they were looking for, or read through a satisfactory amount of it yet. Having to click “Next Page” a dozen times before giving up can be tedious and waiting for each new page to load causes the user to lose interest. This is also the case when the user is not necessarily looking for something, but just reading through information until they’ve felt they’ve read enough. It’s likely that as they read through page after page they will grow tired of waiting for pages to load and having to reload previous pages if they wish to go back.
Where it’s used
Continuous scrolling is used mostly on the web and on game consoles with no mouse-like interaction, but can sometimes be useful for mobile and desktop applications as well.

Why it works
Data is loaded as the user nears the end of the previous chunk of loaded data. That way, once the user has reached the point where they need to see more, it’s already loaded and displayed. This creates a continuous experience for the user, with no wait for pages to load and as much data as desired can be loaded by just scrolling down. It’s likely that the user may scroll through many pages of information without even realizing how much data they’ve actually been through.

How others have used it
Other popular applications that use continuous scrolling include Twitter, Pinterest, Facebook, and Facebook mobile.

Figure 4.1-30: Twitter displays an animated “loading” circle while loading more tweets.

Figure 4.1-31: Pinterest displays a bouncing ball with the label “Fetching pins...”
Continuous scrolling can be found in these libraries/collections:

http://ui-patterns.com/patterns/ContinuousScrolling (figure 4.1-29 source)
http://patternry.com/p=endless-scrolling/
http://designinginterfaces.com/patterns/infinite-list/ (very similar)
http://www.androidpatterns.com/uap_pattern/dynamic-loading-of-a-list

**Continuous scrolling as used in Facebook**

Facebook uses continuous scrolling to present data on the news feed. The user can view stories and as they scroll downward, the page will update to display another chunk, thus expanding the view. If the user scrolls down too quickly, they will see a loading indicator at the bottom of the page (see Figure 4.1-32). This same pattern is used when scrolling down a user’s profile page (Timeline). New posts will display as the user scrolls, with the loading indicator appearing under the last post if the user moves too quickly. The month/year indicator will also update to indicate the time period the user is currently viewing (Figure 4.1-33).

![Figure 4.1-32: Facebook's news feed loading animation.](image-url)
4.1.8 Hover-Reveal Tools  
A.K.A. Hover Controls, Hover Tools

Who it’s for
Any type of application can use this pattern.

What it is
A set of utilities that are related to an object which appear when a user mouses over the object.
Figure 4.1-34: Twitter’s tools for manipulating individual tweets are hover-reveal tools.

When it’s used
It’s not always feasible to have every action visible on-screen. For large lists and collections of items, tools can be redundant and clutter the screen. As user interfaces have evolved, they have moved away from cluttered screen real estate and toward minimalistic views. In order to create a minimalistic environment, buttons and action links can’t be listed on every item. An abundance of dropdown lists can begin to clutter an interface as well, and requires extra involvement for a user to complete a task. The goal is simplicity in both design and interaction, which is difficult to achieve when there are many actions possible.

Where it’s used
Hover-reveal tools are useful on desktop and web platforms.

Why it works
When a user wishes to interact with an object, they simply hover their mouse pointer over that object and contextual tools will appear over it. This allows the user to fluidly select a tool and perform an action without having to click through a separate list of actions or select from a dropdown. Yet, the tools don’t clutter the screen when the user does not have their mouse pointer
over the related object, they just appear as whitespace. This works especially well for lists in which each item has one or more tools that pertain to that specific list item.

How others have used it

Other popular applications that use hover-reveal tools include Google, Facebook, Twitter, and Flickr.

![Google's use of hover tools for search results include +1 and preview (>>) buttons.](http://patternry.com/p-hover-reveal-tools/)

Figure 4.1-35: Google’s use of hover tools for search results include +1 and preview (>>) buttons.

Hover-reveal tools can be found in these libraries/collections:

http://patternry.com/p-hover-reveal-tools/

Hover-reveal tools as used in Facebook

In order to alter a post, a user can mouse over it and select either the star tool, which makes that post a featured item on their Timeline, or the edit tool, which allows the user to change the post’s date, hide the post from their Timeline, add a location to the post, or delete it entirely (figure 4.1-36). Because these options are not seen by the general public, it makes sense to use them as hover-reveal tools so that a user has the same view of their posts as the general public, until they mouse over them. The same principle applies when the user wishes to edit their profile or cover photos. The user simply mouses over the photo and a hover-reveal tool appears as seen in figure 4.1-37. Beyond the Timeline, the news feed also has a less prominent implementation of hover-
reveal tools. If a user mouses over a news feed item, a small dropdown arrow appears and the user can click it to reveal a dropdown list with various actions they can take on that specific news feed item, as shown in figure 4.1-38.

Figure 4.1-36: Facebook’s hover tools for a user’s own Timeline.

Figure 4.1-37: Hover tools used on a user's Facebook Timeline profile picture.
4.1.9 Input Prompt

A.K.A. Labels within Inputs

Who it’s for
Any type of application can use this pattern.

What it is
An instructional label in that is found inside a field and disappears once clicked.

When it’s used
Putting instructions beside or above every field seems necessary so that users know what the field is for, but it takes up a lot of extra space for all of those labels. This is especially wasteful for areas where there is only one or two fields, so once the user knows what they correspond to, they don’t need the space-wasting instruction to be there all the time. Having a lot of these labels can disrupt a page’s layout and make it look cluttered, which is very off-putting to most users.
Where it’s used
The input prompt is equally beneficial on all platforms.

Why it works
Putting the instruction directly into the field space clears up the interface and allows for more information to be displayed without looking cluttered. Many fields on one page can have input prompts, especially if the user is expected to go through them one at a time, like in registration. The user can go to each field, read the prompt and then when they go to enter their information, the prompt disappears and they can type directly into that space. The amount of room on the page can improve immensely and the user sees no loss in instruction or functionality.

How others have used it
Other popular applications that use input prompts include Twitter, Facebook, Apple and Google.

Figure 4.1-40: Twitter’s login screen provides prompts for every field.
The input prompt can be found in these libraries/collections:

http://ui-patterns.com/patterns/InputPrompt
http://patternry.com/p=input-prompt/
http://quince.infragistics.com/#/Search/ViewPattern?pattern=Input+Prompt

**Input prompts as used in Facebook**

Input prompts are used for most of the fields on the Facebook home page. The most-used field on the site is the status update/photo/poll post field, which includes three different input prompts, depending on which type of post the user currently has selected. These can each be seen in figure 4.1-41, this bar can be found on a user’s own Timeline as well. Other places which use an input prompt are the search bar, found in the center of the Facebook header (figure 4.1-41), and the comment prompt on each post or photo/video (figure 4.1-42).

![Facebook's search and status input prompts.](image1)

**Figure 4.1-41:** Facebook’s search and status input prompts.

![Facebook’s comment input prompt.](image2)

**Figure 4.1-42:** Facebook’s comment input prompt.
4.1.10 Overlay  
A.K.A. Lightbox, Floating Window, Lightweight Pop-up

Who it’s for
Any type of application can use this pattern.

What it is
An overlapping message window which is smaller than the main window and normally centered.

![Welcome to Groupon San Francisco!]

Figure 4.1-43: Groupon displays an overlay message to welcome new users.

Screenshot from Patternry.com

When it’s used
To display an important message or to temporarily change activities, a new window is needed to separate the user’s attention from the background window. However, in web applications this is awkward and annoying, not to mention almost impossible now that so many users have pop-up blockers. Navigating away from the page to display the message or activity can also be annoying to users who wish to cancel the action and go back to what they were previously doing.
Where it’s used
Overlays are mostly used on the web but small overlays can be used on mobile devices as well.
Desktop usage is mostly for matching web applications, since normally a desktop application
would just open a new window.

Why it works
The overlay window only momentarily interrupts a user’s activities and can be easily dismissed so
that they can continue what they were doing. When the background is darkened, it immediately
draws attention to the message in the overlay window so importance can be conveyed easily
through overlays. Overlays are not blocked by pop up blockers and can normally be dismissed by
pressing escape or clicking an x to close the box. In some cases, overlays are semi-transparent to
allow users to see that this window is only temporarily blocking the main window.

How others have used it
Other popular applications that use overlays include Facebook, Flickr, Groupon, and Kayak.

Figure 4.1-44: Flicker allows users to view a photo’s location on an overlay map.
Overlays can be found in these libraries/collections:

http://patternry.com/p=overlay/ (figure 4.1-43 source)

**Overlays as used in Facebook**

Facebook uses overlays to allow the user to complete a decentralized action, such as adding a new event, while staying on a related page that focuses on the community and social aspect of that action. In order to add an event from the news feed, the central social hub, a user must first navigate to the events page, which displays other users’ events. Then, they can click the Create Event button to open the Create New Event overlay as seen in figure 4.1-45. Once the event is created, the user is returned to the events page and their new event is displayed in the events list. This allows the user see how their event fits in with the events of the rest of their community- a very important feature of a social networking application.

![Create New Event](image)

**Figure 4.1-45:** Facebook’s event overlay returns the user to the community events page.
4.1.11 Reflector

Who it’s for
Any application which can change the look of a user/page, based on the viewer, can use this pattern.

What it is
A tool which allows the user to view a page they control the content of as another user, or as the general public/guest.

![You will be seen as: photoqueen](image.png)

Figure 4.1-46: An example of a user card reflector by Yahoo!
Screenshot from Yahoo! Design Pattern Library

When it’s used
Because privacy is such a big concern, it’s important for users to feel secure with the amount of data they make available to the public. A user who manages a webpage, such as an online profile, may wish to hide certain pieces of information from other users, or from the general public. Many websites have implemented this feature but because so much information can go into a profile page, and so many options have been made available for hiding and sharing that information, it is difficult for users to sometimes know for sure whether they set those options up the way they wanted to. Because they can’t log in as another user, there is no way for them to know what that particular page looks like to that user if they set up special restrictions for them. This can worry users about the security and privacy of their online data, possibly turning users away completely.

Where it’s used
The reflector is currently used solely on the web, but uses on other platforms may be possible.
Why it works

The reflector allows users to enter in a username and see exactly what their profile page looks like to that person. They can also see what a person who is not logged into an account can see, so as to hide information to the general public. This means that a user who has set up custom privacy options can check to see if the data they want shared or hidden is set correctly, as viewed by a specific person they've included in those options. Seeing exactly what the page looks like to other people gives the user piece of mind and the confidence to use the privacy options at their disposal.

How others have used it

Popular applications that use reflectors include Flickr, Yahoo! Local, and Facebook.

Figure 4.1-47: Facebook’s reflector pane allows viewing by specific friends or the general public.

The reflector can be found in these libraries/collections:


The reflector as used in Facebook

Facebook uses the reflector exactly as it’s intended to be used- to view a user’s profile from the view of another user or guest. Although in recent UI updates the ability to use this tool has been hidden, it can still be found on a user’s profile or Timeline by clicking the settings icon and
selecting “View As...” This option opens a header which allows the user to type in any name on their friends list to see their view of the Timeline (figure 4.1-48). Once a name has been entered, the Timeline refreshes to show that person’s view of it, with a header that reminds the user exactly whose view they are seeing, as well as the ability to change the privacy of each individual post (which is a relatively new feature some users may not know about) (figure 4.1-49).

**Figure 4.1-48:** The View As... button puts the Timeline into a reflector mode.

**Figure 4.1-49:** A Timeline in reflector mode shows the user how it appears to the general public.
4.1.12 Sign-In Continuity

Who it’s for
Any application which requires users to sign in to perform any/all actions (member applications) can use this pattern.

What it is
A prompt for the user to sign in in order to complete a desired action, which returns the user to the action he/she wanted to perform.

![Sign in to view this video](https://via.placeholder.com/150)

**Figure 4.1-50:** YouTube allows sign-in continuity so that users can verify their age.

When it’s used
Many applications (websites in this case) require users to be signed in in order to participate, even if the user can view parts of the site without being signed in. This develops a problem where the user has browsed the content without being signed in, but now wishes to participate in some way and is required to be signed in. Once the user is taken to the sign-in page and successfully signed in, they are often redirected to their designated 'home' page for the website instead of the page they were just on, the one with the content that required them to sign in in the first place. This is frustrating and the user could have a hard time finding the page they were previously on, much less finding it quickly and easily.
Where it’s used
Sign-in continuity is typically found on the web, there is little use for signing in to most desktop, mobile, or other applications.

Why it works
Sign-in continuity allows users to go to a sign in page (or sign in in an overlay window) and then right back to the page they were previously on. It takes very little time to get signed in so that they can perform the task that required it in the first place. Because there is no searching or further navigation required, users are more likely to sign in when they want to participate, knowing that it will be fast and easy.

How others have used it
Other popular applications that use sign-in continuity are Twitter, Facebook, Kayak, and YouTube

Figure 4.1-51: Kayak opens a login overlay (see section 4.1.10) to implement sign-in continuity.
Sign-in Continuity can be found in these libraries/collections:


**Sign-in continuity as used in Facebook**

In a normal case, all users are signed into Facebook if they are already browsing it. However, in the case that a user clicks a link to a product, business, musician, or just a friend’s Facebook page, only public information will be available to them. The user will be prompted to sign in to connect with that person/page; if the user chooses to do so, they can sign in using the top bar (which is always available to users who are not signed in) and the page will refresh to display previously private information. However, using the “Sign Up” or “Log In” buttons will not allow the user to stay on the page, they will be redirected to the Facebook home page (news feed) instead. It would be a good suggestion for Facebook to further implement this design pattern so that logging in with either method uses sign-in continuity.

**Figure 4.1-52:** Visiting the Timeline for TV show Futurama prompts the user to sign in.
4.1.13 Tag

A.K.A. Tag an object

Who it’s for
Any type of application can use this pattern.

What it is
An attribute that may be applied to an object which signifies that the two are related. Normally multiple tags can be applied to any one thing.

![Tags](image)

**Figure 4.1-53:** Vimeo.com allows users to add and remove video tags.

Screenshot from UI-Patterns.com

When it’s used
Many applications have functionality for users to search through the data presented by the app. However, searching for vague terms or not knowing what to search for at all can cause a user’s search results to be unhelpful. The same is true for behind-the-scenes categorization of an application’s data; oftentimes, an application can try to automatically categorize objects such as posts, photos, articles, and more, but it is normally more accurate for a human to do these things. The problem is how to get users to categorize objects on their own.

Where it’s used
The tag can be useful on all platforms, particularly on the web.

Why it works
Tags are easy to add to objects, any user can tag an object with as many words or phrases as they want. Once the objects are tagged, they will appear in the search results for that tag. They may also appear in a tag cloud (a collection of trending tags) which users can interact with to view
popular categories. Applications can also filter a user’s view by certain tags, so say a user searches for articles on a very general topic, they can narrow down the results by filtering to only see articles with a specific tag. Tags are also useful to identify people who are related to the object, such as people in a photo or at a location. Objects normally have a mutual relation with their tags, making it possible for users to view all objects with a specific tag or to view all tags associated with a specific object.

How others have used it
Other popular applications that use tags include Gmail, Amazon, Twitter, Facebook, and CNN.

![Twitter lists trending topics and hash tags by local area.](image)

Tags can be found in these libraries/collections:


Tags as used in Facebook
Users on Facebook can tag photos in an album, videos, comments, and any kind of post such as wall photos, links, and status updates. Many Facebook apps can also use tags. There are two main types of tags on Facebook, the photo/video tag, and the post tag. Photo/video tags identify a user
which is actually in a photo or video. Normally for photos, the user or their friends can tag others by selecting a face on the image and typing the associated name into the search box (figure 4.1-55). Then when others view the photo, hovering over the tag will show the box around that person’s face (or other tagged region) as seen in figure 4.1-56 or hovering over the person’s face will show their name on the photo (figure 4.1-57). Tags are also listed next to the photo description. The post tag is used by typing @ and a friend’s name as seen in figure 4.1-58 or by clicking the add friend button and searching for the friend the user wishes to tag. That person’s name will be shown as a clickable link once tagged, either in the status (by using the @ method) or after the status (by using the button). For both types of tags, a post will be added to the tagged user’s Timeline and a notification will be sent to that user. For photo tags, the photo (or group of photos if they were from the same album) will be shown on the tagged user’s Timeline. For tags in a post, that post will show up in its entirety on the Timeline of every user tagged in it. Although Facebook’s tags are actual users of the website, the mutual relationship of tag and object still applies in certain cases. Users can view all photos or videos with a certain person’s tag by going to that person’s profile and viewing their photos or videos as seen in figure 4.1-59.

Figure 4.1-55: Adding a tag to a photo.

Figure 4.1-56: Hovering over a tag to display the tagged region.
Figure 4.1-57 (Left): Hovering over the photo region to display the tag.

Figure 4.1-58 (Above): Adding a tag using the @ symbol.

Figure 4.1-59: Sorting all Facebook album photos based on one user's tag.
4.1.14 Terms of Service Button

Who it’s for

Applications who require users to accept a terms of service agreement can use this pattern.

What it is

A button which explicitly states that clicking it is akin to accepting the application’s terms of service.

Figure 4.1-60: An example of a Terms of Service button by Yahoo!. Screenshot from the Yahoo! Design Pattern Library

When it’s used

Applications of all kinds have a terms of service agreement that they often require a customer to read and accept before using the application. When installing, or first using an application, users don’t often take the time to look over the setup carefully and can easily miss a checkbox stating “I have read and agree to the terms of service.” Users who don’t want to carefully look over the application’s installation try and click through the steps quickly, and are disrupted by a deactivated continue button or an error preventing the user from continuing. It is not always immediately obvious why the progress has stopped and the user must then find the unchecked checkbox (which may be off-screen) and check it before continuing.
Where it’s used
The terms of service button is equally beneficial on all platforms.

Why it works
Terms of Service buttons put the agreement right under the user’s mouse pointer. Users who speed through setup won’t be stopped by other means of acceptance, nor will they have to scroll down pages upon pages to get to the acceptance checkbox. Users who have read the terms of service can indicate that they agree by continuing, no extra buttons required. The entire process is streamlined for quick, easy acceptance so that the user can get to using the application as soon as possible.

How others have used it
Other popular applications that use terms of service buttons are PayPal, Facebook, traditional Mac installers. These installers often replace the standard “I agree” checkbox with an extra dialog that pops up when the user tries to continue past the terms of service. The dialog simply asks if the user agrees to the terms of service and presents an agree button and a disagree button (which will cancel setup) and essentially simulates a terms of service button.

Figure 4.1-61: A standard Mac installer prompts the user to agree to the terms of service.
The terms of service button can be found in these libraries/collections:

http://developer.yahoo.com/ypatterns/social/people/engagement/tos.html (figure 4.1-60 source)

**The terms of service button as used in Facebook**

When registering for a Facebook account, the user must accept the terms of service agreement first. Facebook has a quick and easy account registration and allows the user to input the minimal amount of information in the page, and click the button that says Sign Up. Above the button, a note states “By clicking Sign Up, you agree to our Terms and that you have read and understand our Data Use Policy, including our Cookie Use.” with links to the terms of service, data use policy, and cookie use policy.

![Facebook Sign Up Page](image)

**Figure 4.1-62:** Facebook’s sign up page uses a terms of service button indicated in the fine print.
4.1.15 User Card

A.K.A. Hover Card, Identity Card

Who it’s for
Any type of application can use this pattern.

What it is
A small pop-up panel which normally appears when a user hovers over a target's name or avatar.
The panel contains a photo of the target, their full name, and any related information.

Figure 4.1-63: LinkedIn’s version of a user card
Screenshot from Patternry.com

When it’s used
Users often want to know more about a target without straying from their current activity. It breaks the user's train of thought to have to leave the page to learn more about another user (or similar target) and can be frustrating when only basic information is desired. This is especially true for email applications when the user wants to send mail to a specific person but only has a list of email addresses to choose from; it’s too tedious for the user to have to look up each email address in their contacts to see the full name of the person it belongs to.

Where it’s used
The user card is normally found on the web but desktop usage is also possible.

Why it works
User cards are convenient because they only require that a user hover over a target in order to see its basic information. If the user recognizes a name but wants to see more about that person or
organization, they can hover over it (or in select cases, click it) to see a small photo and possible a phone number and address. Likewise, hovering over an avatar will display the information of the user who the avatar represents, possibly with a larger avatar image in the user card. This lets the user learn more about targets without leaving the page or breaking away from the current activity. Phone numbers, email addresses, addresses, and other basic information is easily accessible and the user wastes no time to view it.

How others have used it
Other popular applications that use user cards include Gmail, Yahoo!, LinkedIn, Facebook, and Digg.

Figure 4.1-64: Digg’s user cards for a source.
Screenshot from Patternry.com

Figure 4.1-65: Gmail’s chat user card.

Tags can be found in these libraries/collections:

http://patternry.com/p=user-card/ (figure 4.1-63 and figure 4.1-64 source)

User cards as used in Facebook
Whenever a user posts a photo album, a status update, a comment, or a link, their profile picture and name are displayed on the post, other users can hover over it to see the user card. Another place it’s used is when a user is tagged in a post, other users can mouse over the tag to see the
tagged user’s card. If they are friends with the tagged user, they will normally see their profile picture, name, mutual friends and whatever basic information (like workplace and school) that user has entered (figure 4.1-66). If the user is not friends with that person, they can only see what that person has allowed the general public to see (from their privacy settings); by default, this includes their profile picture and mutual friends. There is a button on the bottom of each user card that shows the friend status between the users, this can include Friends, + Add Friend (not yet friends) and Friend Requested. At the introduction of friend groups, Facebook altered this to show a checkmark in front of the word friends if they are only friends, and a star in front of the word if they were in another group such as Close Friends (figure 4.1-67). This adds functionality to the user card, in addition to displaying basic information.

Figure 4.1-66: Facebook's user card displayed by hovering over a user's status update.

Figure 4.1-67: The user card of a friend in a special friend group.
Section 4.2 Interaction Patterns from the Facebook Mobile Apps

As users rely more and more on their mobile devices, it’s important for all major applications to have a mobile app for users on-the-go. Facebook has an app in both the Android Marketplace (Now called Google Play) for android phones and tablets and also in the iTunes Store, for iPhones, iPads, and iPods. These applications have been downloaded by users worldwide and are good examples of integrated mobile applications using a similar user interfaces as the web version of the application. In this section, we study the interaction patterns of the mobile applications specifically. This is an upcoming part of the field, as the introduction of smart phones was only a few years back.

The following patterns will be presented in this section:

4.2.1 – Carousel

4.2.2 – Expandable List

4.2.3 – Fling to Scroll

4.2.4 – Mode Selector

4.2.5 – Pull to Refresh
4.2.1 Carousel

Who it’s for
Any application which contains a series of images can use this pattern.

What it is
A way to view multiple image items in a row, by circling through each image and its neighbors.

![Figure 4.2-1: The Android carousel pattern from AndroidPatterns.com](image)

When it’s used
Displaying images in an album setting, where the user must load a new image in one at a time is too time-consuming. Users must wait for the next image to load each time and there is no good way to indicate how far into the album the user is currently at. For mobile devices, this means that in addition to long loading times there must also be buttons on screen for forward and backward navigation through the album. This takes up valuable screen space; the user wants the images they are viewing to be maximized on small devices and these navigation buttons take up this space.
Where it’s used
The carousel is useful on all platforms.

Why it works
Laying images out in a carousel pattern allows the user to view the previous and next image, as well as the one they are currently centralized on. New images load while the user is viewing previously loaded ones so that wait time is minimized. For mobile devices, users can flip through images using screen interactions such as flinging or sliding to change the image. The pattern is dynamic so the user can go forward and backward as much as desired and also view parts of images without switching to the full image focus.

How others have used it
Other popular applications that use the carousel include Amazon, Yahoo!, Facebook, and Facebook mobile.

Figure 4.2-2: Amazon’s mobile suggestions are listed in a carousel.
The carousel can be found in these libraries/collections:

http://www.androidpatterns.com/uap_pattern/carousel (figure 4.2-1 source)
http://ui-patterns.com/patterns/Carousel

The carousel as used in Facebook mobile
Both the mobile and web versions of Facebook use a carousel to view photos in an album. On the
web, users may click small arrows to the left or right of a photo to navigate to the next one. The
mobile version is even more useful, users can slide their finger across the screen in the direction
they’d like to “turn” the carousel and view the next image. Because all of the images are loaded
before the user moves to the next image, they can move back and forth between images without
delay. The photo caption/description is listed at the top of the screen and switches to whichever
photo is taking up more than 50% of the screen, the photo tools are available at the bottom of the
screen so that the real estate in the center is used only by the photos. For photos which have the
same orientation as the screen (portrait/landscape) the top and bottom areas will appear over top
of the photo with a transparent background. The following three screenshots show the movement
of the carousel in landscape mode between photos in a photo album. Screenshots are from an
Android device but are similar to Apple app.
**Figure 4.2-3:** Between the first two photos in the carousel.

**Figure 4.2-4:** Focused on the second photo in the carousel.

**Figure 4.2-5:** After moving past the second photo in the carousel.
4.2.2 Expandable List  A.K.A Mobile Accordion List

Who it’s for
Any application which contains a (long) list can use this pattern.

What it is
A list which includes list items which are expandable to display more items, an alternate to continuous scrolling.

Figure 4.2-6: The Android expandable list pattern from AndroidPatterns.com.

When it’s used
When a user needs to view a long list of items, it used to be customary to implement lists with pagination. However, users had to wait for each page to load once they clicked the button to continue to another page, even if they were going backward to data that had already been loaded once before. On a mobile device, this could be especially annoying for a user to have to wait on so longer lists are typically used. Depending on the list size and contents, a longer list could take a long time to load initially, therefore the same problem arises. Continuous scrolling (see section 4.1.7) was the solution to this type of problem but it is not always appropriate, especially if the data being loaded is segmented into parts or should be able to be collapsed (similar to the accordion lists in section 4.1.1) to save space.
Where it’s used
The expandable list is useful on all platforms, although typically used on mobile devices.

Why it works
When dealing with long lists of items that are able to be categorized or segmented, it’s beneficial to group them into expandable subheadings which act as the new list. So when users view the expandable list, they see the subheadings and can choose the one that best matches what they want to view or can just expand each of them to see the full list. This allows users to skip over data that is not relevant to them, which saves time in loading that data and in picking through irrelevant list items, as well as saving screen real estate while the subheadings are collapsed. Categorizing list items by date is very common, displaying recent entries first (possibly by default) and allowing later time periods to be expanded and collapsed.

How others have used it
Other popular applications that use the expandable list include Android browsers, Facebook mobile, and Microsoft Outlook.

Figure 4.2-7: An android browser’s history before being expanded.
(Note this only applies to portrait mode).

Screenshot from AndroidPatterns.com

Figure 4.2-8: An Android browser’s history after being expanded.

Screenshot from AndroidPatterns.com
The expandable list can be found in these libraries/collections:

http://www.androidpatterns.com/uap_pattern/list-expandable-list (screenshots source)

**The expandable list as used in Facebook mobile**

The Facebook mobile Timeline displays the most recent month’s worth of information, such as posts and recent activity. If a user scrolls down to view earlier posts, they will see an expandable list which allows them to select from the next month’s worth of information, or by year as seen in figure 4.2-9. Once a time period is expanded, it cannot be collapsed, but the subheading remains separating the information between time periods as seen in figure 4.2-10. This pattern is not used on the Facebook mobile news feed or on the iPhone app, continuous scrolling is implemented there (as discussed in section 4.1.7).

![Figure 4.2-9: The bottom of the current month on a Facebook Timeline, before being expanded.](image)

![Figure 4.2-10: The Timeline after April has been expanded.](image)
4.2.3 Fling to Scroll

Who it’s for
Any touch application which contains a long list of items.

What it is
The gesture of flinging (sliding one’s finger across the screen and then picking it up) the screen scrolls the page/list in the direction that the user moved their finger and slows to a stop once released.

![Image of Fling to Scroll gesture]

Figure 4.2-11: The Android fling to scroll pattern from AndroidPatterns.com.

When it’s used
Navigation in a mobile application adapts to the type and amount of information displayed on any one page/view. Slide to scroll options stop the movement of the list as soon as a user lifts their finger from the screen; this is useful for precision applications but tedious for users to scroll through a very long list. The only distance they can scroll at one time is the length of the screen, then the user must lift their finger and start at the bottom again. This can take a long time and frustrate the user.
Where it’s used
Fling to scroll is used on mobile platforms, it is also possible to be implemented on other devices with touchscreens.

Why it works
Allowing the screen to continue moving after the user has lifted their finger means that it can travel further without the user having to manually move the screen. To scroll faster a user just moves their finger faster in the direction they would like to go. To stop the screen immediately they can just press their finger against the screen without moving it and the screen will stop. This allows for the precision of slide to scroll with extra speed and with less work for the user.

Although it is not possible to really demonstrate fling to scroll through screenshots, one feature of it can normally be seen in them- the scroll bar. A small scroll indicator normally appears while the user is scrolling, to show them their location in the context of the list; this indicator disappears after the user stops scrolling (see figures 4.2-12 and 4.2-13).

How others have used it
Other popular applications that use fling to scroll include the mobile versions of Twitter, Facebook, Google+ and Amazon.
Fling to scroll can be found in these libraries/collections:

http://www.androidpatterns.com/uap_pattern/fling-to-scroll (figure 4.2-11 source)

Fling to scroll as used in Facebook mobile

All of Facebook’s mobile application uses fling to scroll. This is especially useful because there is so much data to scroll through in the Facebook app. Whether on the news feed, a Timeline, or a user's friends list, fling to scroll helps the user to move quickly long lists. Once the user sees what they are looking for and wishes to stop scrolling, even while moving very quickly, they can tap the screen to stop the scrolling. Unlike many other applications, Facebook does not use a scroll bar indicator on the Android app (but does on the iPhone app), which may be a good suggestion for future user interface improvements. The following screenshots are of the Facebook news feed in both landscape and portrait mode on an android device. Fling to scroll can be used no matter what the screen orientation and although normally used for vertical lists, it can also be used for horizontal lists (possibly similar to the carousel in section 4.2.1).
4.2.4 Mode Selector

Who it’s for

Any application which contains a filterable list or a list with multiple views can use this pattern.
What it is
A button on a list view that allows a user to select between multiple modes to change the contents of the list.

![Figure 4.2-16: The Android mode selector pattern from AndroidPatterns.com.]

When it's used
With the limited screen space on most mobile devices, it’s very important to maximize screen real estate. A filter button, or dropdown is often needed in order to let the user customize the contents of a display, yet the user normally wants the list to take up as much of the screen as possible to reduce the amount of scrolling needed. This presents the problem of where to place such an option, and how to minimize its impact on the interface. The tool must also reflect what mode is currently selected, and must travel with the list as it scrolls so that the user can choose to change modes without scrolling to the beginning of the list.

Where it’s used
The mode selector is useful on mobile, web, and desktop platforms.

Why it works
The mode selector is a dropdown list button which dynamically changes its appearance to display the currently selected mode (which makes it an indicator as well as a button). Normally anchored
to the application’s mobile header, the mode selector often overlaps the top of the list in a way that does not obscure data but does not uselessly take up space as well. As the user scrolls down the list, the header and anchored mode selector stay at the top of the screen. This allows the user to access the dropdown easily if they choose to change modes, it also keeps the selector in one centralized position. Most mobile apps already have a header which functions in this manner, so the mode selector is simply an extension of it for lists. The contents of the dropdown are not static but rather, they can change based on which list is being viewed; normally, the dropdown options are displayed in an overlay window, but possibly in a new screen.

How others have used it

Other popular applications that use the mode selector include the mobile versions of Twitter, Facebook, Time Magazine, and Engadget.

The mode selector can be found in these libraries/collections:

http://www.androidpatterns.com/uap_pattern/mode-selector (screenshots source)
The mode selector as used in Facebook

Facebook already has a header bar across the top of their mobile applications. The mode indicator is simply included in that when viewing the news feed, the only filterable list on the Facebook mobile app. The symbol displayed on the button indicates which mode is currently in use and an arrow next to it prompts the user to click to see more options (figure 4.2.19). There are two modes that can be selected from the dropdown, they appear in an overlay window as seen in figure 4.2.20. These options filter the posts on the news feed by popularity (based on Facebook’s ranking algorithm) or by time. If Top Stories is selected, a newspaper icon is shown; if Most Recent is selected, a clock icon is shown. As the user scrolls down, the header stays the same so that the user can easily filter their news feed at any time. The iPhone app’s mode selector does not indicate graphically which mode is selected. It simply has a button labeled “sort” instead. This could be a good future improvement to this app.
4.2.5 Pull To Refresh

Who it's for

Any touch application which has real-time data changes.
What it is
A method of refreshing the page by moving the screen in a specific manner instead of pressing a button.

Figure 4.2-21: The Android pull to refresh pattern from AndroidPatterns.com.

When it’s used
Like in all other mobile user interface designs, screen real estate is of the utmost importance.

When a list contains data that may change before the list updates, there must be a way to refresh the page manually. Placing a button on the page takes up valuable space and may not ever be used by the standard user. Placing the refresh option in an options or context menu makes it more difficult for users who are trying to refresh the page for the first time to be able to find the option. No standard way of refreshing had been implemented (prior to this design pattern) so users most always had to hunt for the button to refresh the page or wait until the list refreshed automatically, which is a waste of time and frustrating for a user who wants to see current data.

Where it’s used
Pull to refresh is used on mobile platforms, it is also possible to be implemented on other devices with touchscreens.
Why it works
When users scroll up to the top of the list to see if the new data has been refreshed, a message displays that tells the user to pull down to refresh, once the user drags the message downward it instructs the user to release to refresh the page and then updates the page’s information. There are no buttons to find and many applications already use this form of refreshing, so once users discover it once, which is easy to do even by accident, they will be able to easily refresh the page with a flick of the hand. The reason that the users must first pull down the refresh area is that scrolling up past the top of a page is quite a common occurrence when fling to scroll is implemented and it safeguards the user from accidentally refreshing the page. A refresh button is often still included in the options menu in case a user does not know about this method. The lack of a dedicated refresh button on the screen saves space, so more information can be displayed on the page. The ease of refreshing the page with this method makes it the most natural option for implementing a manual refresh option.

How others have used it
Other popular applications that use pull to refresh include the mobile versions of Twitter and Facebook.

Figure 4.2-22: Twitter's first pull to refresh message.

Figure 4.2-23: Twitter's second message for the user to release to refresh.
Pull to refresh can be found in these libraries/collections:

http://www.androidpatterns.com/uap_pattern/pull-to-refresh-2 (figure 4.2-21 source)

**Pull to refresh as used in Facebook**

Facebook uses pull to refresh (on both Android and Apple devices) in a typical manner on each user's Timeline. To refresh the content, a user must scroll up past their cover photo and then pull the instruction downward as seen in figure 4.2-25. The user releases after the arrow indicator changes to up as shown in figure 4.2-26 and then the newest data will be loaded into the Timeline (figure 4.2-27). A page can be refreshed as many times as a user wishes, whether there is new information waiting to be loaded or not.
Figure 4.2-27: Successful refresh of the page displays a spinning indicator during the refresh.
Chapter 5  Conclusion and Future Work

We’ve seen many of the various interaction patterns implemented by Facebook applications. It’s apparent that each of these design patterns has a vital purpose in the user experience. Because each of these patterns is used in other popular applications, users can adjust quickly to interface changes instead of having to re-learn the application from scratch. This makes new applications easy to use even without extensive instruction. Facebook’s Help website is able to answer any questions a new user may have and also inform advanced users of the more detailed actions available. Using patterns in user interface design also helps the designer to produce a fluent, up-to-date user interface. Each interaction pattern itself also changes with the times. As new needs are introduced, interaction patterns must evolve to meet these needs. Separate from the interaction patterns themselves, as the look and feel of applications changes, the interaction patterns can remain the same, only the examples will seem dated. This can be seen when looking at the older interaction patterns from Jennifer Tidwell’s book Designing Interfaces from 2005. Example screenshots have square designs, similar to older Windows operating system styles. Newer screenshots in her 2011 edition include more rounded styles with sleek designs and more stylish color schemes, but the pattern is the same. Because design patterns can outlast the toll of time, they are a confident way to design user interfaces.

In the future, it’s possible that we will see one pattern library rise up and become the standard reference for design patterns. The pattern layout presented in this thesis (who, what, when, where, why, how) may become the new standard template for interaction patterns. Interaction patterns may also be integrated with object-oriented patterns to create a pseudocode library for all of the interaction patterns. It would be beneficial for a central layout, library, or pattern collection to be created so that patterns can be consistent. Most assuredly, Facebook will continue to be on the cutting edge of user interface design and more design patterns will come from
Facebook applications, as well as other popular applications on the market today. Some of the suggestions made in this thesis will probably be implemented later on, as Facebook’s interface designers update the web and mobile applications. Some of the applications mentioned in this thesis will become obsolete and others will rise to take their place. We can only hope that design patterns become a more active part of software design, especially in user interfaces, so that we will have sleek, easy-to-use, yet functional interfaces for years to come.
Bibliography


Screenshots and descriptions reference these applications:
(In order of first appearance)

Microsoft Outlook Web – http://www.live.com
Facebook – http://www.facebook.com
YouTube – http://www.youtube.com
Gmail – http://mail.google.com
Windows 7 – Microsoft Windows 7 Ultimate
Amazon – http://www.amazon.com
Ebay – http://www.ebay.com
Travelocity – http://www.travelocity.com
Twitter – http://www.twitter.com
Pinterest – http://www.pinterest.com
Google – http://www.google.com
Flickr – http://www.flickr.com
Kayak – http://www.kayak.com
Mac Installer – Netbeans 7.1 installer

All web screenshots taken using Mozilla Firefox.
All mobile screenshots taken from a Motorola DROID Bionic.