Anchored Customization

Anchoring Settings to the Application Interface to Afford Customization

by

Antoine Ponsard

B.Sc., École Polytechnique, 2012
Diplôme d'Ingénieur, École Polytechnique, 2013

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF
THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE

in

The Faculty of Graduate and Postdoctoral Studies

(Computer Science)

THE UNIVERSITY OF BRITISH COLUMBIA

(Vancouver)

November 2015

Abstract

The HCI community has identified the need to let users adapt their software to their own tasks and preferences. Yet, many users do not customize, or only do so rarely. The de facto standard customization mechanism is the settings panel, which has undergone minimal design improvements since it was introduced along with the graphical user interface in the 1980s. Entirely disconnected from the application UI, these panels afford only very indirect manipulation, as users must rely on often cryptic text labels to identify the settings they want to change. From a *developer's* point of view these panels make sense: they are simple graphical representations of traditional UNIX config files.

In this thesis, we propose a novel customization approach, designed from the *user's* point of view. In Anchored Customization, settings are anchored to *conceptually related* elements of the application UI. Our Customization Layer instantiates this approach: users can see which UI elements are customizable, and access their associated settings. We designed three variants of customization layer based on multi-layered interfaces, and implemented these variants on top of a popular web application for task management, Wunderlist. A remote experiment conducted on Mechanical Turk, complemented with a face-to-face lab experiment (for a total of 60 participants) showed that the two minimalist variants were 35% faster than Wunderlist's settings panel. This new approach provides significant benefits for users while requiring little extra work from designers and developers of applications.

Preface

The experiments described in this thesis were conducted with the approval of the UBC Behavioural Research Ethics Board (certificate number H11-01976).

Parts of this thesis appear in a conference paper manuscript [1] where I am the first author. Joanna McGrenere helped frame and write the manuscript, and provided supervisory assistance for this research.

[1] A. Ponsard, J. McGrenere. Anchored Customization: Anchoring Settings to the Application Interface to Afford Customization

Table of Contents

Abstract ii

Preface iii

Table of Contents iv

List of Tables vii

List of Figures viii

Acknowledgments ix

Dedication x

1 Introduction 1
- 1.1 Anchored Customization 1
- 1.2 Contributions 2
- 1.3 Overview 3

2 Related Work 4
- 2.1 Adaptable Interfaces 4
- 2.2 Adaptive and Mixed-Initiative Interfaces 4
- 2.3 Customization in Context 5
- 2.4 Interface Context 5

3 Systematic Assessment of Feasibility 6
- 3.1 Existing Customization Mechanisms 6
- 3.2 Feasibility of Anchored Customization 7

4 Design 9
- 4.1 Anchored Customization 9
  - 4.1.1 Display of Settings 13
  - 4.1.2 Display of Anchors 13

4.2 Customization Layer . . . 14
4.2.1 Anchors Visible in a Layer . . . 14
4.2.2 Three Variants for Displaying Settings . . . 15
4.2.3 Search . . . 18
4.2.4 Software Architecture . . . 20
4.2.5 Implementation . . . 20

5 Evaluation . . . 21
5.1 Experiment 1: Remote Mechanical Turk . . . 22
5.1.1 Method . . . 22
5.1.2 MTurk Results . . . 25
5.2 Experiment 2: Face-to-Face in the Lab . . . 26
5.2.1 Method Differences from Study 1 . . . 27
5.2.2 Face-to-Face Lab Results . . . 27
5.3 Secondary Exploratory Analyses . . . 31

6 Discussion . . . 34
6.1 Reconciling the Performance Results . . . 34
6.2 Performance/Awareness Trade-Off . . . 35
6.3 Applicability to Other Desktop Software . . . 35
6.4 Applicability to Handheld Devices . . . 36
6.5 Interpretation Gulf . . . 36
6.6 The Developers' Point of View . . . 36
6.7 Limitations . . . 37

7 Conclusions and Future Work . . . 39
7.1 Long-Term Effects . . . 39
7.2 Generating the Mapping . . . 40
7.3 Concluding Remarks . . . 40

Bibliography . . . 42

Appendices

A Mapping of settings to UI elements in Wunderlist . . . 47

B Experiment 1 Resources . . . 50
B.1 Tasks . . . 50
B.2 Participant Consent Form . . . 53
B.3 Questionnaires . . . 56

B.3.1 Demographics 56
B.3.2 Satisfaction 56
B.3.3 Recognition 56

C Experiment 2 Resources 59
C.1 Tasks 59
C.2 Call For Participation 59
C.3 Participant Consent Form 61
C.4 Questionnaires 64
C.4.1 Demographics 64
C.4.2 Ratings 66
C.4.3 Rankings 66

List of Tables

3.1 Summary of a systematic review of 8 PTM apps to determine the feasibility of Anchored Customization 7

List of Figures

4.1 Three approaches to organize settings: config files, settings panels, Anchored Customization 10
4.2 Two approaches to navigate settings: tabs and Anchored Customization 11
4.3 The settings panel offered in Wunderlist 12
4.4 The Wunderlist web application 16
4.5 Customization Layer displayed on top of Wunderlist 16
4.6 Ghost anchors and collapsing mechanism 17
4.7 Minimal, Minimal+Context, and Full+Highlight variants 17
4.8 Reverse highlighting in the full settings panel 19
5.1 Modal dialog displayed at the beginning of each trial 23
5.2 Possible states of the progress bar, displayed at the top of the window during the experiment 24
5.3 Experiment 1: 95% confidence intervals of the median duration per Customization Mechanism and Block 26
5.4 Experiment 2: 95% confidence intervals of the median duration per Customization Mechanism 28
5.5 Density plots of the distribution of the percentage of trials in which participants selected a “correct” anchor 32
5.6 Density plots of the distribution of the percentage of trials in which participants selected a “correct” anchor, for each Customization Layer variant 32

Acknowledgments

I thank my supervisor Joanna McGrenere, for her endless patience, guidance and support throughout this Masters.

LUNCH, MUX, and HCI@UBC were valuable occasions to meet fellow HCI researchers at UBC. I thank all the people who organized these meetings, and all those who provided feedback on my work. In particular, I thank Francisco Escalona, Kamyar Ardekani and Kailun Zhang, with whom I worked on three great projects. I also thank Peter Beshai, Derek Cormier, Mona Haraty, Matthew Brehmer and Oliver Schneider for their advice and the many insightful discussions we had.

I thank my Information Visualization professor, Tamara Munzner, who courageously accepted to be my second reader.

Finally, I thank my family and friends for their encouragement and support.

This research was funded by the Graphics Animation and New Media (GRAND) NCE as well as the NSERC Discovery Grant program.

Dedication

There is one thing that is more dear to a Frenchman's heart than the sight of the Eiffel tower over the rooftops of Paris: a properly baked loaf of bread. Alas, this simple pleasure is surprisingly difficult to find outside of the grassy plains of France.

This is why I would like to dedicate this thesis to the bakery Beyond Bread, for their remarkable effort to propagate the baking traditions of France to the distant Canada. As soon as I set foot in this bakery, the smell of freshly baked bread washing over me, I knew that I had found a home away from home. I cannot be thankful enough for their country bread, with its firm crust and delicate crumb; for their properly named *pains au chocolat*; for their rosemary foccacia, soft as a dream and dripping olive oil; and even for their overpriced baguette and croissants.

This work would not have been possible without them.

Chapter 1

Introduction

The HCI community has identified the potential benefits of supporting users to adapt their software to their own tasks and preferences [28]. Yet, many users do not customize, or only do so rarely. In today's software, the de facto standard customization mechanism is the settings panel, also known as the "options menu" or "preferences dialog." These panels have significant usability limitations in that the settings they offer are entirely disconnected from the application UI. There are no visual affordances in the UI that can help answer the question: "Is it possible to customize X?" Users must open the settings panel and then rely entirely on text labels to identify the settings they want to change, with little to no other contextual information. Thus, the classic vocabulary problem applies: the language used by designers to describe an aspect of the interface is often different from the language of users [19]. This creates a deep gulf of execution between the user's intentions, formulated in the context of the application UI, and the actions offered in the settings panel. The fact that customization occurs fairly infrequently over long periods of time [27] only exacerbates the problem: users are likely to forget where things are in the settings panel between customization episodes.

Despite obvious usability problems, settings panels have undergone only minor improvements since they were first introduced along with the graphical user interface in the 1980s [24]. The interaction paradigm fundamentally hasn't changed: the user can browse tabs of settings, tick checkboxes, and choose values from drop-downs. These panels essentially represent a *developer's* point of view of the application's customization opportunities: they are simple graphical representations of traditional UNIX config files.[2]

1.1 Anchored Customization

We propose a novel approach to customization, called Anchored Customization, designed from the *user's* point of view: *anchoring* settings to visual elements of the UI that are *conceptually related* to these settings. This reduces the gulf of

[2]Config files list in plain text the settings that can be modified. Users can change settings by opening and editing the file in a text editor.

execution, and leverages users' existing knowledge of the application UI. While others have proposed the idea of moving the access point of a customization closer to the feature that it affects [38], we take the approach further by allowing users to access all conceptually related functions from an anchor point. For instance, a notification icon in the UI can be used as the anchor to change not only *how* popup notifications (emanating from the icon) are displayed, but also the *frequency* of various notifications (e.g., email), and other related settings. Anchored Customization leverages *mental associations* that users can form intuitively between visual elements of the interface and changes they want to make to their software.

Once settings are anchored to UI elements, there are different possible ways of visualizing and accessing them. We designed, implemented, and evaluated Customization Layer, as a concrete instantiation of the Anchored Customization approach. Customization opportunities are represented in a meta-layer on top of the interface. Users can see all UI anchors at once in this layer, and can access their associated settings by clicking on them. This type of layered design has been successfully used in the context of online help [8]. Although clicking on an anchor only shows a small subset of settings initially, the user can access the full settings panel from any anchor via a simple expansion mechanism.

Our approach is pragmatic: it recognizes that settings panels are the standard in the software industry. Developers rely on this simple mechanism to provide end-user customization, since these panels are (presumably) an inexpensive part of the UI to develop. And users are accustomed to them. Introducing a new customization mechanism therefore requires careful consideration of the cost/benefit trade-offs at play. We show that anchoring settings to the UI through a customization layer has the potential to provide significant benefits for users, while requiring only minimal extra work from the designers/developers. Our goal is to propose a well-considered interface improvement that can be adopted widely.

A customization mechanism must be implemented to customize a specific app. Although our design approach is applicable to any software with a GUI that is customizable via a settings panel, we had to choose a particular application domain for a first evaluation. Since task management tools are widely used, and significant individual differences have been observed in the way people manage their tasks [22], we considered that Personal Tasks Management (PTM) would be an appropriate first application domain for our research.

1.2 Contributions

Our work contributes the following:

1. The concept of Anchored Customization: to anchor customization opportunities to conceptually related elements in the application UI.

2. A systematic review of PTM apps to evaluate a priori the feasibility and potential usefulness of adopting this concept.

3. A realization of this concept in a customization mechanism, the Customization Layer, with three variants that explore different trade-offs between multi-layered interfaces.

4. An implementation architecture of these three variants for web applications: our prototype augments a real-world PTM app, Wunderlist [20], but can be adapted to other modern web apps with minimal additional work for developers and designers.

5. Two experiments (Mechanical Turk and face-to-face) with 60 participants showing performance improvements of the Customization Layer over the regular settings panel provided by Wunderlist. The face-to-face experiment also provided a preliminary understanding of how users apprehend Anchored Customization.

1.3 Overview

Previous work relevant to this research is summarized in Chapter 2. Chapter 3 presents two systematic reviews of a set of Personal Task Management apps. The first review identifies the customization mechanisms provided in these apps; the second evaluates the feasibility of anchoring settings to visual elements of the interface. Chapter 4 outlines the design space of Anchored Customization, and explains the design of our Customization Layer. Chapter 5 presents a two-part evaluation of our Customization Layer, in comparison to traditional settings panels. A first experiment was conducted remotely on Amazon Mechanical Turk, to measure the performance of our Customization Layer for changing settings. A second experiment took place in a face-to-face settings at our university, to verify and triangulate the results of the first one. Chapter 6 discusses the insights gathered on the three variants of our Customization Layer, and reflects on the possibilities offered by this new customization mechanism. Chapter 7 outlines directions for future work and concludes this thesis.

Chapter 2

Related Work

Previous research has studied the benefits of letting users adapt software to their own preferences and tasks [27, 28, 30]. What exactly constitutes customization is not well-established in the HCI community, despite there being a rich literature in this space. For example, Mackay [27] defines end-user customization as "the mechanisms by which users may specify individual preferences, and preserve their preferred patterns of use, without writing code." In that sense, it is closely related to the concept of tailorability, in which "end users can progressively modify a working system without ever having to leave the application domain to work in a separate underlying programming domain." [28]. Yet the term "tailorable" seems somewhat outdated in the HCI community, and has been replaced by "adaptable".

2.1 Adaptable Interfaces

The challenges of designing and building *adaptable interfaces* has produced a rich and varied literature. Multi-layered interfaces [30, 33] group subset of features into different layers, and allow users to choose which layer to work with. Sophisticated techniques have been proposed to let users modify GUIs at run time [12, 13], even without access to the source code [11, 31, 36]. The end-user development paradigm goes even further, by empowering users to build their own applications [17, 26]. Yet, powerful adaptable approaches generally require users to spend significant time and effort personalizing their interfaces, often by writing scripts or editing code—which limits the potential audience to power users. As a result, these approaches have not yet been widely adopted in industry.

2.2 Adaptive and Mixed-Initiative Interfaces

Adaptive approaches attempt to reduce the need for user involvment by automatically generating user models that predict potentially useful customizations [16]. However, giving control of the user interface to an automated system has significant drawbacks, as it tends to make the UI spatially unstable and unpredictable [14]. *Mixed-initiative approaches* can alleviate those problems by suggesting possible

customizations to the user, who remains in control of their interface [7, 10]. Ultimately, adaptive and mixed-initiative approaches put the onus on developers to build and integrate complex user models into the interface, which is difficult to do well [23].

2.3 Customization in Context

Our work focuses on leveraging the customization opportunities already available in existing software settings panels, with the goal of minimizing the extra costs of customization for both users and developers. The potential of "representing tailoring functions in the overall interface" was identified 15 years ago [25], but has received little attention since then. An early work proposed the concept of Direct Activation [38] which states that "the access point of the tailoring function should be designed related to the one of the tailorable function". In other words, the settings that affect a given feature should be accessible from (or around) the access point of the feature they affect. We broaden this approach by anchoring any setting to any conceptually related elements of the UI —not only elements objectively affected by a setting. For instance, the setting for changing the keyboard shortcut used to "open a new tab" could be anchored to the button that opens a new tab. This setting does not tailor the "open a new tab" function itself, but conceptually it makes sense to associate a keyboard shortcut and a button that call the same function.

2.4 Interface Context

The benefits of preserving *interface context* have been exploited in other areas of HCI. Perhaps the most common example is the context menu, which reduces the time needed to select options related to the current location of the pointer. For example, contextual help can augment an application interface with links to a wiki [35], or user-generated Q&As [8]. While help queries tend to naturally correspond to elements of the interface that users are trying to learn, the correspondence between settings and interface elements may be less clear, as settings can affect software in arbitrary ways. Thus, it was not obvious at the outset of our research that such direct correspondences could be found from settings to UI elements. Yet the literature offers many promising examples of representing *meta-information* about an application as an overlay on the app interface: recent changes [2, 5], computational wear [29], predicted use [18], even low-level usability problems [37]. This motivated our visual design approach, which considers customization opportunities as a meta-layer on top of the application interface.

Chapter 3

Systematic Assessment of Feasibility

The motivation of Anchored Customization is to address the shortcomings of settings panels, which appear to be the standard in the software industry. To verify this assumption, we performed a systematic review of a set of PTM apps, investigating which customization mechanisms they provide. We then conducted a second review to evaluate the feasibility of the Anchored Customization approach.

3.1 Existing Customization Mechanisms

We chose a particular yet fairly generic application domain: Personal Task Management. We selected 12 of the most popular PTM apps[3], for a total of 20 unique apps (counting separately both desktop and Android versions). For each application we recorded:

- which customization mechanisms were offered;
- how many user actions were needed to access them;
- the location of the access points to these mechanisms in the application interface;
- the type and number of settings each mechanism could change.

Our review showed that an overwhelming majority of settings were accessed via settings panels. In fact, only 1 of the 20 applications didn't offer a settings panel at review time—but it now does. The *structure* of these panels was remarkably similar: tabs for desktops apps, multi-level menus on Android. In both cases, additional visual delimiters such as lines or boxes were sometimes used inside each tab or level, to create subsections of related settings. The same widgets were used to change settings: checkboxes or toggles for binary settings, drop-downs

[3]Evernote, OneNote, Any.do, Cal, Wunderlist, Todoist, Remember The Milk, Toodledoo, Astrid, Clear, GTasks, and Tasks.

Application	Platform	# settings	directly	indirectly	not
Astrid	Android	41	25	6	10
Evernote	Desktop	31	11	12	8
Gmail	Web	29	16	7	6
RTM	Android	26	10	11	5
Toodledo	Web	53	37	7	9
Toodledo	Android	43	26	16	1
Wunderlist	Web	41	12	28	1
Word	Desktop	132	90	24	18
total		396	227	111	58
normalized %		100%	52%	32%	16%

Table 3.1: Summary of a systematic review of 8 PTM apps to determine the feasibility of Anchored Customization. For each app, the settings provided in the settings panel were classified as directly, indirectly, or not anchorable. To avoid biasing the results towards the applications offering the most settings, we normalized the different proportions of settings for each app before averaging across all apps.

or radio buttons for multiple choices, and in some cases sliders for numbers. The access points to these settings panels were also very similar: an "Options", "Preferences" or "Settings" menu entry at the top of the screen, or a gear or toolbox icon.

We found that settings panels also contain a variety of non-settings items, such as an "About" page, contact information, or promotional materials for upgrading to a paid version. In that sense, they serve somewhat as a mixture of miscellaneous elements that were probably not deemed important enough to be integrated in the main application interface.

We did encounter a few other customization mechanisms, such as dragging a border to resize a column, or reordering buttons via drag-and-drop. These alternative mechanisms effectively use direct manipulation to let users customize, but were typically restricted to changing a very specific part of the interface.

3.2 Feasibility of Anchored Customization

The widespread use of settings panels highlights the potential *usefulness* of Anchored Customization. We next evaluated the *feasibility* of this approach on a subset of apps from our first review. We selected apps with more than 15 settings, and included Microsoft Word and Gmail which, while being general-purpose soft-

ware, can both also be used for task management. As shown in Table 3.1, we reviewed a total 8 apps (5 on desktop, 3 on Android), classifying a total of 396 settings.

According to our estimation, 52% of settings could be anchored unambiguously in the interface. A large number of these correspond to settings that directly show or hide interface elements, or change their position or visual appearance. Another 32% of settings could be *indirectly* anchored based on an intuitive mental association. For instance, the frequency of automatic syncing could be anchored to the button that performs a manual synchronization. Finally, we estimated that only 16% of settings could not be anchored anywhere in a meaningful way. These settings correspond either to advanced options (e.g., number of weeks after which completed tasks are archived), or global options (e.g., display language) which are not related to any particular UI location.

The percentages reported here are based on the raw number of settings offered by the apps we reviewed. However, not all settings are created equal: some will be changed more often than others. A more accurate statistic would be to weigh each setting by how often users can be expected to change it. We believe that this adjusted metric would reduce the relative proportion of non-anchorable settings even further. Indeed, only a small fraction of users (probably those with more technical backgrounds) is likely to dig into the most advanced settings.

Overall, the results of our second review provide solid evidence for the viability of our Anchored Customization approach, at least in the PTM domain: approximately 84% of settings can be anchored. Yet a significant number of settings cannot be anchored (16%), which must be taken into account when designing anchored customization mechanisms.

Chapter 4

Design

In most applications, settings are a list of parameters that can take some predefined values. They are generally given a human readable label, and sometimes a short description. For example, a "confirm before deleting" setting can be either true or false, and could have the description "show ok/cancel popup when clicking on the delete button". UNIX-type config files are the most barebone representation of settings, and the closest to the programming domain (Figure 4.1 a). Users can manually change settings by editing the config file in a text editor.

Settings panels offer two key improvements over config files. First, they prevent errors by restricting the values that a given setting can take. For instance, checkboxes only toggle booleans, and drop-downs offer a limited set of options to choose from. Second, settings panels often group related settings together into tabs or other forms of subsections, with the aim to reduce the time needed to access a particular setting. In this way, settings panels are a *static abstract partition* of settings: each setting appears in only one section; sections are based on abstract categories, such as "shortcuts" or "display;" and these categories are defined by designers at the outset (Figure 4.1 b). Tabs are used to navigate from one high-level category to another (Figure 4.2 a). An example is shown in Figure 4.3.

4.1 Anchored Customization

The Anchored Customization approach, by contrast, promotes a *contextual many-to-many mapping* of settings to UI elements (Figure 4.1 c). Any UI element can be used as an *anchor*—for instance, icons, buttons, menus, even an empty area. The goal of the mapping is to provide context for each setting. One setting can be mapped to multiple anchors, and multiple settings can be mapped to the same anchor. The visibility of anchors and thus their associated settings changes dynamically, depending on the current state of the interface. The Anchored Customization approach leverages users' pre-existing knowledge of the application UI, instead of requiring them to learn the abstract structure of a settings panel. Hence, the key idea of Anchored Customization is that the application interface itself is used to organize and navigate the settings space (Figure 4.2 b).

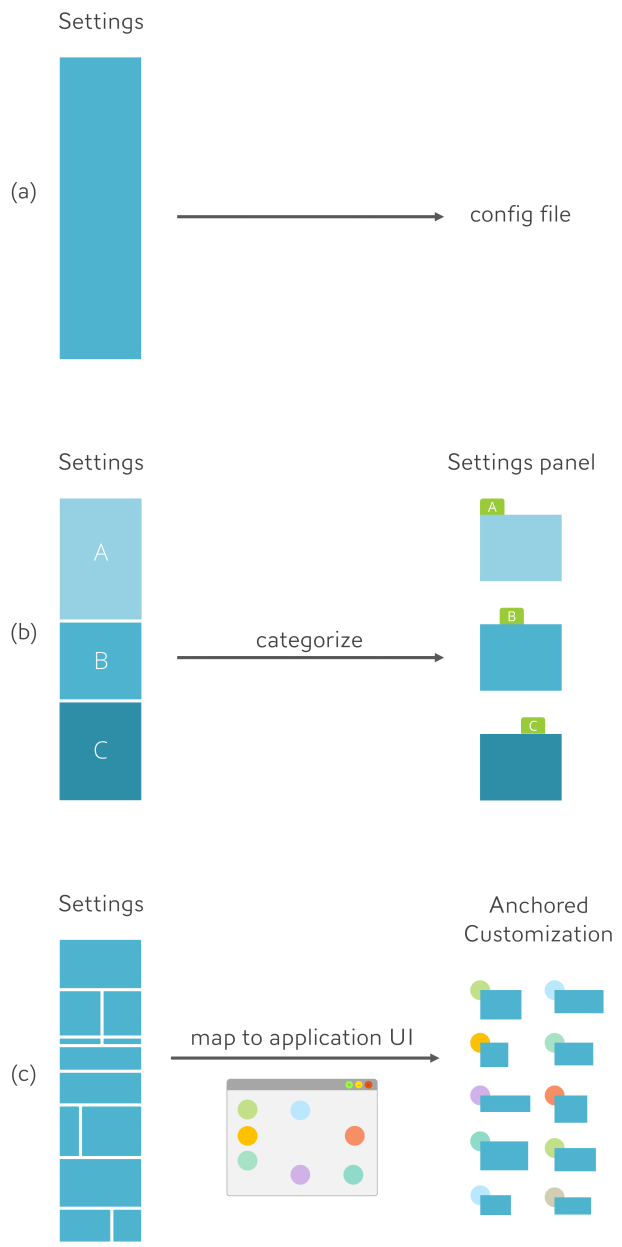


Figure 4.1: (a) Config files are the most direct representation of settings, without any particular structure. (b) Settings panels are a static partition of settings, based on abstract categories. (c) Anchored Customization creates a contextual many-to-many mapping of settings to UI elements. UI elements are represented here by colored circles.

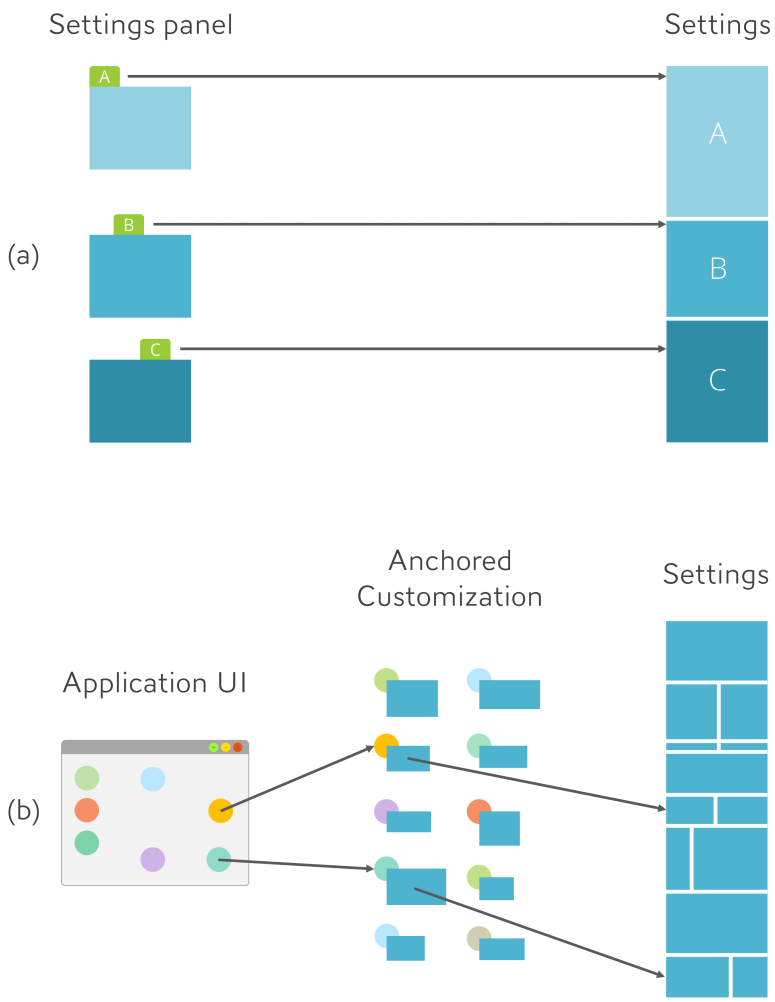


Figure 4.2: (a) In a settings panel, tabs indicate the current location and allow users to navigate to a different group of settings. (b) In Anchored Customization, elements of the application UI itself are used to navigate the settings space.

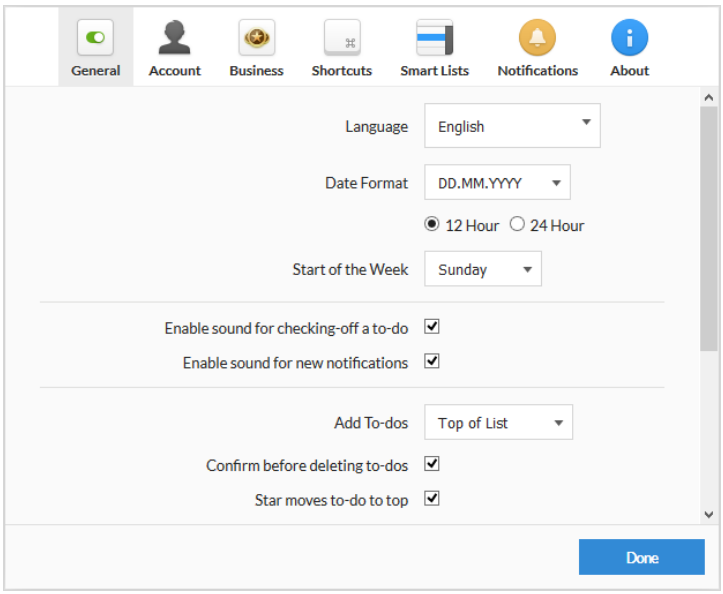


Figure 4.3: The settings panel offered in Wunderlist, which was used as the Control condition in Experiments 1 and 2. Tabs appear at the top of the panel, and are decorated with an icon.

There are two main dimensions in the design space for mechanisms that instantiate Anchored Customization: the display of the settings and the display of the anchors.

4.1.1 Display of Settings

It is not desirable to permanently show the settings associated with the various anchors. Most applications offer many settings, which would clutter the interface if they were always visible. Further, customization is typically very much a secondary infrequent activity. Thus, in general, the settings associated with an anchor should be displayed *on demand*, when the user expresses interest in an anchor—by clicking or hovering on it, for instance. There are many ways to display settings once demanded. We explored three possibilities, inspired by multi-layered interfaces (described below). But the design space is much larger. Some interesting dimensions include:

- *Which* settings should be displayed when clicking on an anchor. Only the settings mapped to this anchor, or a local neighborhood?
- How to *represent* settings. Now that settings are placed in context within the interface, the text labels might become superfluous. For instance, shortcuts might not need a textual description if they are mapped clearly to a button that performs the same action. Settings that show or hide an element might be better represented by a simple visible/invisible icon. In general, different *types* of settings (or their associated anchors) could be represented differently.
- How to provide access to the settings that are *not associated* with any anchor.
- How much of the *structure* of traditional settings panels should be retained, to help users transition to this new approach to customization.

4.1.2 Display of Anchors

There are a number of different possibilities when considering how the anchors can be displayed. One familiar approach is through context menus: anchors are not explicitly marked visually, but can be used by the user to demand a corresponding setting. For instance, in Microsoft Word, right-clicking the ribbon shows a contextual menu with an option to "Customize the ribbon". A tooltip that appears when hovering over an anchor for a short duration could be used in a similar way. This local, targeted approach answers the question: Is *this* customizable?

The downside of this approach, however, is that there is no way for the user to get a holistic overview of all the settings available within the UI. Serially invoking the context menu from many different anchors is too tedious, and the user must resort to bringing up the settings panel.

An alternative approach is to provide a *global overview* of all the customization opportunities available, by making all the anchors visible at once, likely through a mode. This approach helps answer the question: *What* is customizable? The downside of this approach is that entering a distinct mode could interrupt the user's workflow —which is why modes should be used sparingly in interaction design. The contextual menu approach described above might be more appropriate when users can easily guess (or already know) which UI element to click on to perform the desired change. In contrast, the global overview approach helps users to become aware of the settings available, and where to find them. Fortunately, the two approaches are not mutually exclusive; providing both would allow users to choose the one best suited to their current need.

4.2 Customization Layer

We designed and implemented Customization Layer which instantiates the anchored customization approach. In the design dimensions described above, this mechanism uses a *global overview* approach to show all anchors in an extra *layer* on top of the interface. We explore three ways to display settings on demand, inspired by multi-layered interfaces.

4.2.1 Anchors Visible in a Layer

In a layered design, there are multiple ways to indicate which UI elements are anchors. In LemonAid's contextual help mode [9], the interface was overlaid with yellow question marks that explicitly represent help queries. To avoid clutter, we thought it best not to add additional graphical elements to mark anchors. Instead, the anchors themselves are visually highlighted when users activate a layer on top of the regular application (shown in Figure 4.4). In our implementation, the app interface becomes darker and less saturated, but is still clearly visible through the Customization Layer. Anchors are shown with a white background and dark gray text, to optimize legibility and ensure visibility above the dimmed application interface (Figure 4.5).

When the user hovers over an anchor A, that anchor and all other anchors mapped to the same settings as A are highlighted in orange (Figure 4.5). This linked highlighting helps users create a mental model of the mapping of settings to UI elements. For instance, in a PTM app, hovering over a button highlights all

the buttons that share the same setting (e.g., a keyboard shortcut for marking a "todo" as important).

There is one important special case that required attention when anchoring settings to UI elements: what if a setting governs the visibility of *the UI anchor itself?* Consider for instance settings that show or hide buttons in a toolbar. Of course it makes sense to anchor such a setting to the button it affects; but after hiding the button, how can users access its associated setting to make the button visible again? To avoid this problem, we introduce the notion of *ghost anchors:* anchors that correspond to hidden elements in the application UI. These anchors offer the same functionality as regular anchors, but are displayed in a darker gray to indicate their transient status.

Displaying all ghost anchors at the same time is not necessarily desirable. If an application has many optional, hideable components, showing all of them as ghost anchors could break the application layout, clutter the customization layer, and overwhelm the user. Furthermore, the customization layer should look as similar as possible to the current interface of the application, to help users alternate seamlessly between the two. For these reasons, we implemented a collapsing mechanism: ghost anchors are by default represented by a simple "chevron" icon in the customization layer (Figure 4.6). To further reduce visual clutter, ghost anchors that are close to each other are collapsed under the same chevron icon, based on a simple proximity clustering algorithm. Clicking on a chevron icon reveals the ghosts that it was previously hiding.

4.2.2 Three Variants for Displaying Settings

When the Customization Layer is shown, clicking on an anchor displays the subset of settings anchored to that UI element. We explored three visual representations of this subset (see Figure 4.7). The Minimal panel (M) only shows the settings in the subset, with an orange background to indicate that they are related to the anchor that was clicked. Minimal+Context (M+C) is very similar, but a tab name next to each setting indicates from which tab of the full settings panel it comes from. In contrast, Full+Highlight (F+H) keeps the structure of the complete settings panel, and the settings from the anchor's subset are highlighted in orange, as are the tabs in which they appear. Some tabs contain more settings than can be shown in the height of the panel, so we automatically scroll down to bring into view the first highlighted setting, if any.

Because some settings might not be anchorable to the interface (cf. Feasibility section), all designs must provide a fallback access to the complete set of settings. In Full+Highlight, users can browse all the tabs freely, as they would in a regular settings panel. In Minimal+Context, the tab names next to each setting are ac-

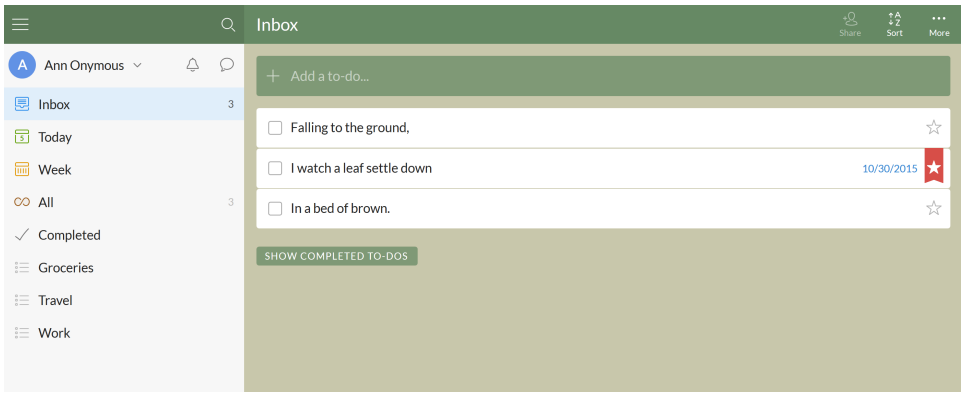


Figure 4.4: The Wunderlist web application. Todos appear in a vertical list, and can be "starred" to convey importance. Here the user has starred the second todo and given it a due date. In the left sidebar are a collection of "Smart Lists", intelligent filters that display a list of todos filtered by a particular criteria. For instance, a Smart List can show only the todos due today, or due this week, or all the todos already completed. There are also user-defined todo lists, such as Groceries, Travel, and Work.

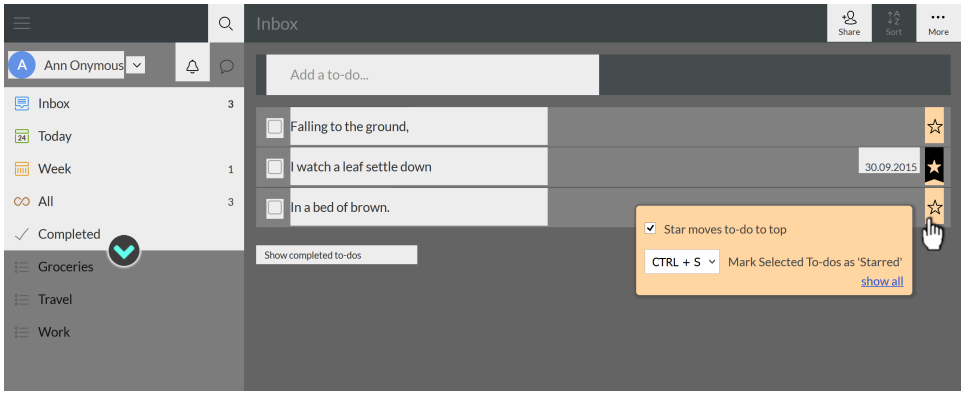


Figure 4.5: Customization Layer displayed on top of Wunderlist. The user is currently hovering over the bottom "star" button on the right-hand side of the screen; as a result, all star buttons are highlighted in orange because they share the same settings. Clicking on an anchor displays the settings associated with it—here, shown in the Minimal panel.

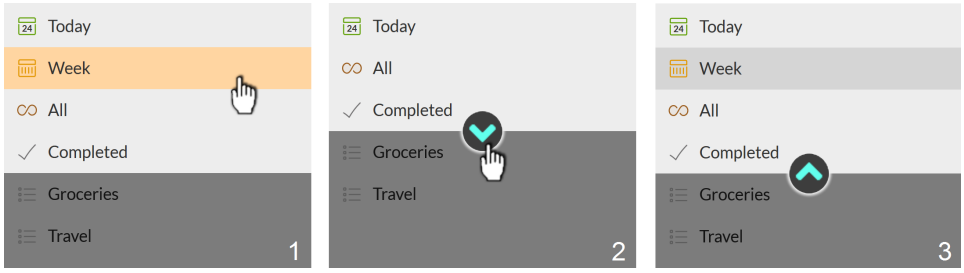


Figure 4.6: (1) The user selects the "Week" filter, then disables it (not shown). (2) The corresponding anchor is collapsed under a chevron icon. (3) Clicking on the chevron reveals the "Week" filter as a ghost anchor.

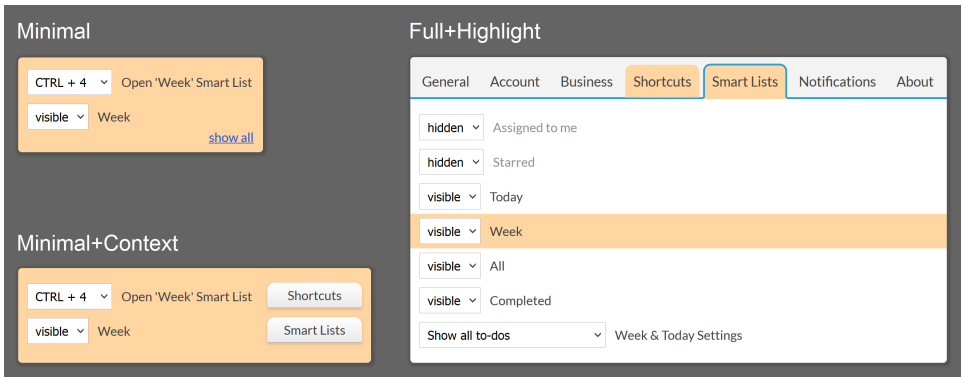


Figure 4.7: Minimal, Minimal+Context, and Full+Highlight variants showing the settings anchored to the "Week" Smart List. Two settings are mapped to this Smart List: the first changes the shortcut for opening this Smart List; the second determines whether this Smart List is visible or hidden in the main interface.

tual buttons; clicking on them opens the full panel at the corresponding tab. In Minimal, a "show all" hyperlink is provided at the bottom of the mini panel. By default, it opens the full panel at the tab that contains the most settings from the anchor's subset. Users can return to the minimal panel by clicking a backward arrow at the top left of the full panel.

These three variants reflect different points in the multi-layered interface design space [33]. The two minimal variants only show the anchor's subset of the settings initially, while Full+Highlight uses visual highlighting to distinguish the subset from all other settings. All three designs are intended (to varying degrees) to reduce complexity, speed up access to the most relevant settings, and help the user become aware of the full set of settings. Hence, they reflect different tradeoffs between a minimal subset of settings and the full settings panel.

As with any multi-layered interface, transitioning from a lower layer to an upper layer can be challenging for users [33]. In our implementation, we provide animated transitions between the minimal variants and the full panel to help users understand the relationship between the two. The minimal panels expand smoothly to the full size, while the highlighted settings glide into their new position. The remaining (non-highlighted) settings fade in afterwards. Informal pilots found these relatively simple animations to be helpful for conveying the intended mental model.

As an additional way to help users understand the mapping between settings and anchors, we augmented the full settings panel with *reverse highlighting:* hovering on any setting displays a blue halo around the corresponding anchors in the interface (Figure 4.8). The idea is to allow users to explore the settings → anchors relationship, whereas clicking on anchors and seeing anchors being highlighted in orange always follows the anchors → settings direction.

4.2.3 Search

Text search is another effective approach for accessing settings. It essentially provides a way to "jump" to a particular setting without having to locate an anchor or browse the settings space. Search is readily afforded by traditional config files, but rarely offered in settings panels—except in complex software such as Eclipse. The downside with search is that the vocabulary problem [19] applies, as users can only guess search terms [4]. Furthermore, relying on search may hinder users' ability to learn other settings. Although not yet implemented, Customization Layer is compatible with text search: UI anchors could be filtered out if their associated settings don't match the search query, or visually highlighted if they do.

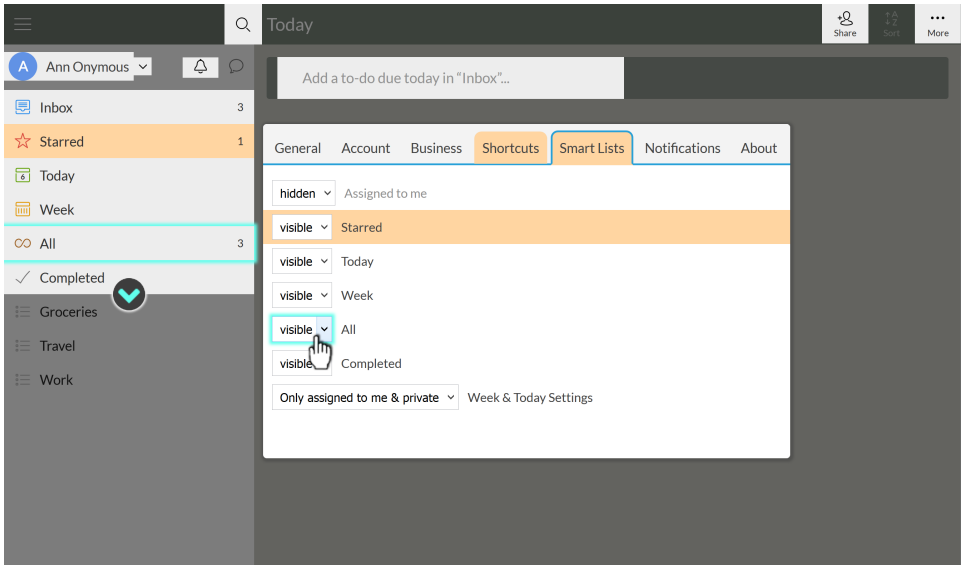


Figure 4.8: The user has clicked on the "Starred" Smart List anchor. As a result, both the anchor and its associated settings are highlighted in orange. The user is now hovering on a different setting in the full panel—the visibility of the "All" Smart List. This setting and all the anchors associated with it are "reverse highlighted" with a blue halo to create a visual relationship between them.

4.2.4 Software Architecture

Our software architecture was designed to be app-independent and extensible to other web apps. Chapter 5 explains why we chose web apps, and Wunderlist in particular, as a concrete starting point for design and evaluation. Adding our Customization Layer to a web app requires: (1) an API to read and write the settings, and (2) a mapping between settings and visual elements of the interface, provided by the designers of the app. UI elements are represented in the mapping by CSS selectors, which allows among other things selecting many similar anchors via a single CSS class (Appendix A). Since designers are already familiar with the settings of their app, creating the mapping should take at most a few hours, except for very large applications.

4.2.5 Implementation

Since Wunderlist doesn't provide an API to its settings, we reverse-engineered its front-end to access the underlying settings directly. Fortunately the settings themselves had human-readable names, which allowed us to localize and modify them programmatically. We then manually mapped these settings to appropriate elements of the interface, which took about two hours. From this mapping, our code automatically generates anchors by creating copies of the DOM elements matched by the CSS selectors provided. Anchors are then positioned precisely on top of the original element, creating the illusion that the elements themselves are highlighted.

When users click on an anchor, we used Angular.js [21], a front-end JavaScript framework, to dynamically filter the list of settings shown in Minimal and Minimal+Context, and to highlight the appropriate settings in Full+Highlight. The smooth transitions between the minimal and full panels were not straightforward to implement, since the start and end positions must be computed in advance for each setting, before the animation starts.

We also prototyped a simple point-and-click web tool that could help designers without web programming skills to retrieve CSS selectors for the UI elements they are pointing at.

Chapter 5

Evaluation

We focus our evaluation on assessing the usability of our Customization Layer. Given the novelty of this new mechanism, the two experiments presented here were intended to be exploratory. We certainly hoped that Customization Layer would help participants find settings faster than browsing a traditional settings panel; but which of the three variants would be the fastest or most preferred was unknown. Further, we anticipated that the three variants would impact users' awareness of the full set of settings differently: the more contextual information a design provides, the greater the user's awareness should be [15].

Choice of the Application

Although our design approach is applicable to any software with a GUI and customizable via a settings panel, we had to choose a particular application on which to evaluate it. We considered building a generic PTM app from the common features and visual styles we observed in our systematic application review (Section 3). The benefit of building a generic app is increased generalizability, since the particularities of each app are averaged out. However, we would have had to hand-pick the customization settings available in this generic app, a process that could introduce some bias in the experiment. Ultimately, we thought it best to evaluate Customization Layer within an existing application with its actual settings. This choice maximizes the ecological validity of our experiment.

We chose Wunderlist [20], one of the most popular PTM apps today (over 10 million downloads on Android and iPhone) which offers a well-designed web interface. Of the 20 apps from our systematic review, Wunderlist had a particularly clean settings panel, with numerous and varied settings. It therefore appeared to be a good baseline for a fair comparison. Furthermore, we were able to access Wunderlist's settings directly on the web client, effectively bypassing the settings panel to customize the app in real time.

5.1 Experiment 1: Remote Mechanical Turk

The primary goal of this experiment was to evaluate the performance of our Customization Layer for changing settings, compared to a traditional settings panel. We used a between-subject design to avoid negative carry-over effects, because switching back-and-forth between two very different mental models could have been confusing for participants. We deployed this experiment on Amazon Mechanical Turk (MTurk), an online crowdsourcing platform that facilitates the recruitment of a large number of participants.

5.1.1 Method

Participants

All 48 participants (aged 19-60, median 28.5, 16 females) were regular computer users, and none had tried Wunderlist before. We had replaced 3 participants, who were either 2.5 Inter-Quartile Range slower than others in their condition, or did 2 IQR more errors than everyone else.

Task

The experiment consisted of a sequence of settings changes. At the beginning of each trial, a popup instructed participants to change one setting to a given value (Figure 5.1). Pressing a "Go!" button would close the popup and start the timer. The instructions were written in layman's terms, and did not necessarily use the same words as the label of the target setting (Appendix B.1). For example, the instruction "Change shortcut for checking off todos" applied to the setting "Mark selected do-dos as completed" in the Shortcuts tab. To help ensure that participants read the instructions before starting the trial, the "Go!" button was kept inactive for the amount of time required to read the instructions at an average reading speed. During that time, the settings panel (in Control) or the UI anchors (in Customization Layer) were hidden, to prevent participants from planning their actions before the timer was started.

During the trial itself the instructions were visible at the top of the window, in case participants had forgotten them (Figure 5.2). Participants had to change at least one setting before the "Next" button became available in the bar at the top of the screen. Clicking this button would stop the timer, indicate whether the task had been successfully completed, and move on to the next trial. We enforced a 2-minute time limit for each trial, after which the trial was marked as a timeout and participants were taken to the next trial. If participants made mistakes, the

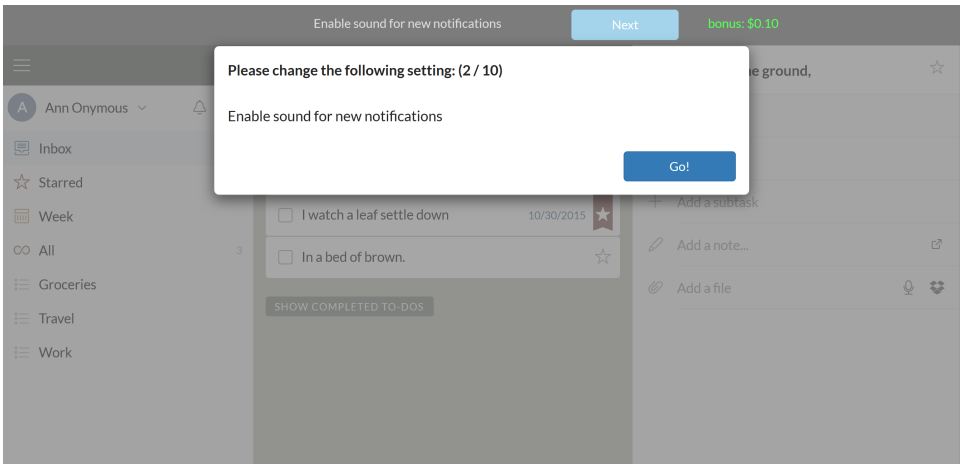


Figure 5.1: Modal dialog displayed at the beginning of each trial. The instructions are repeated in a persistent progress bar at the top of the window, which also shows the bonus reward earned so far. Neither the settings panel nor the UI anchors are visible until participants press the "Go!" button to start the trial.

settings that had been changed incorrectly were reset at the end of the trial, and the target setting changed to the correct value.

Our Customization Layer and the settings panel used in the Control Condition are both accessed via the menu at the top-left of the screen, by clicking on "Customize" or "Settings" respectively. Since activating the customization mechanism takes exactly the same amount of time in both cases, we didn't include this tedious step in the experiment, and instead opened the appropriate customization mechanism automatically in all conditions.

Measures

The duration of a trial was measured between the time when participants pressed the "Go!" button and when they changed a setting. At the end of the experiment, participants completed two recognition questionnaires (Appendix B.3.3) to assess incidental learning. The first was a list of 10 tab names: 5 that appeared in the Wunderlist settings panel, and 5 that we made up to closely resemble actual tab names. Participants were asked to indicate for each tab whether or not it appeared in Wunderlist. Each correct answer was worth 1 point, each incorrect answer -1, so that answering at random would result in a score of zero. The second questionnaire was similarly a list of 20 settings: 5 that participants had changed, 5 that were present in Wunderlist but that participants hadn't changed, and 10 made up

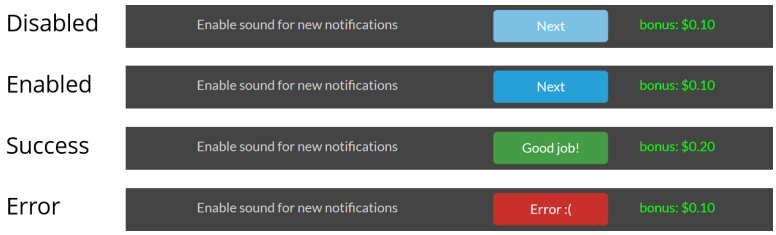


Figure 5.2: Possible states of the progress bar, displayed at the top of the window. The "Next" button is disabled until participants change at least one setting. After changing a setting, participants must click this button to end the trial. If the correct setting was changed, a positive feedback is given and the bonus reward is incremented. Otherwise, participants are told that they made an error and are taken to the next trial.

but plausible. The scoring was identical to the first questionnaire. Finally, participants were asked to rate their satisfaction on a 7-point Likert scale: the extent to which they liked or disliked the customization mechanism they were using, and how easy it was to find the settings they were looking for (Appendix B.3.2).

Conditions

We compared four customization mechanisms: the three variants of Customization Layer (M, M+C, F+H) and the actual settings panel offered by Wunderlist (Control), shown in Figure 4.3. While we accurately reproduced the structure of Wunderlist's panel in our F+H prototype, the visual style was slightly different: Wunderlist generally looks more polished, with custom drop-downs and an icon on each tab.

Wunderlist currently offers 41 settings in four tabs[4]. Out of these settings, we discarded the application language one, as changing it would make the interface indecipherable for our participants. The experiment tasks were not equally difficult: some settings were indeed harder to find than others, especially if their text label was unclear. To reduce the variability between subjects, we partitioned the settings into two groups of 20, carefully chosen to balance the type and location of the settings they offered. Each participant was assigned one set. To assess any early learning, participants were asked to change the same group of 20 settings twice, but in a different randomized order for each of the blocks of 20.

The experiment required changing settings from a default value to a specified

[4]Three other tabs show non-settings information, such as "upgrade your account" and "about this product." We did include these tabs in our prototype.

one. It was not clear *a priori* whether the default value of a setting would impact the ability of users to find and change that setting. For instance, un-ticking a checkbox corresponds to negating its text label, which might be less intuitive than ticking it, following the original meaning of the label. Therefore, we controlled the default values of the settings: participants either started the experiment with the defaults offered by Wunderlist, or the opposite of these defaults.

Design

A 2-factor mixed design was used: 4 customization mechanisms (M, M+C, F+H, Control; between-subjects) x 2 blocks (within-subject). There were two control variables, fully counterbalanced between participants: group of settings (group 1, group 2) and default settings (normal, opposite). Each participant completed 2 blocks of 20 trials, for a total of 1920 trials across all 48 participants.

Procedure

After accepting the HIT (work assignment in MTurk), participants were asked to create a temporary account on Wunderlist, using a randomly-generated email address. Then participants had to drag a custom bookmarklet (a "bookmark applet") to their bookmarks bar, and to click on it to load the experiment code in the Wunderlist webapp. A series of popups walked participants through a quick tutorial, demonstrating the most useful features of Wunderlist. Participants were then given one practice trial, before starting any trial. In the 3 Customization Layer conditions, the practice trial contained a few additional instructions on how to use this new customization mechanism. Finally, participants were asked to complete the two recognition tests, to rate the customization mechanism they were using, and to complete a simple demographics questionnaire (Appendix B.3.1). The whole procedure lasted 32 minutes on average, and participants received an average compensation of $4.34, depending on their performance in the trials and recognition questionnaires. (Incentivizing participants with a variable bonus reward is a standard practice for MTurk studies.)

5.1.2 MTurk Results

We ran a mixed-design ANOVA on the duration of trials. The data was log-transformed to satisfy the assumption of normality, and we used medians to reduce the influence of outliers. The effect sizes reported are generalized eta-square (η^2_G), interpreted as follows: .02, .13, .26 for a small, medium, and large effect size respectively [1]. As shown on Figure 5.3, there was a main effect of customization mechanism ($F_{3,44} = 3.58$, $p < .05$, $\eta^2_G = 0.17$), as well as block ($F_{1,44} = 124.63$,

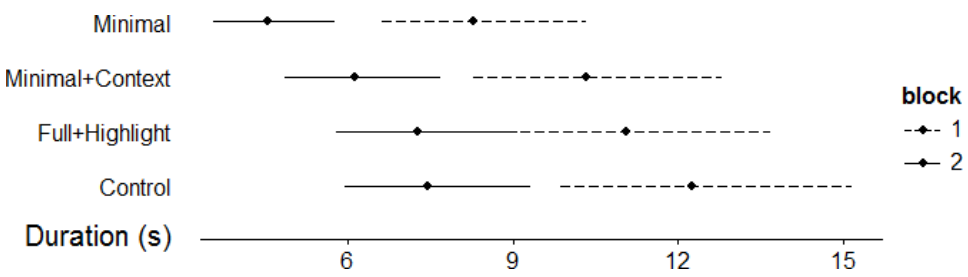


Figure 5.3: Experiment 1: 95% confidence intervals of the median duration per Customization Mechanism and Block. M is significantly faster than F+H and Control, but not significantly different from M+C. (N=48)

$p < .001$, $\eta^2_G = 0.32$), but no interaction between the two. M was significantly faster than Control ($p < .05$, 35% faster) and F+H ($p < .05$, 30% faster). No other pair was significantly different. All pairwise comparisons use a Bonferroni correction unless otherwise mentioned. As anticipated, there was no effect of the two control variables (the group of settings and default settings used). There were few timeouts and errors across all conditions (20 and 54, respectively, for 1920 trials), and they had no significant effect on our results.

A single-factor ANOVA on the tabs recognition scores found a significant effect of customization mechanism ($F_{3,44} = 5.82$, $p < .01$, $\eta^2_G = 0.28$). M scored lower than both F+H ($p = .05$) and Control ($p < .05$), by 2 out of 10 points lower on average. No other pair was significantly different. There was no difference either in the recognition scores for settings, nor in the subjective ratings provided at the end of the experiment. The median rating for ease of use and satisfaction was the same for each of the four customization mechanisms: 2 on a scale ranging from -3 to +3.

5.2 Experiment 2: Face-to-Face in the Lab

Although MTurk provided quick access to a diverse set of participants at a moderate cost, conducting remote experiments has limitations in terms of the insights and qualitative feedback that can be gathered. Thus, we ran a second experiment in a face-to-face lab setting. This time the customization mechanism was treated as a within-subject factor, to allow participants to make informed comparative judgments.

5.2.1 Method Differences from Study 1

We recruited 12 participants (age 21-42, median 24.5, 5 females), all regular computer users. Two had tried Wunderlist before, but were not using it regularly. The experiment task was identical, except that participants were encouraged to talk aloud. The experiment fit in a single one-hour session, for which participants received a $10 compensation. Contrary to MTurk, there was not variable monetary incentive based on performance.

A single-factor within-subject design was used, with the same four customization mechanisms. Participants had to change 10 different settings with each mechanism, using all the 40 settings available in Wunderlist (Appendix C.1), for a total of 12 x 40 = 480 trials. All participants started with the same defaults settings, since the previous experiment did not find any effect of this control factor. The three Customization Layer variants were blocked together, to limit the number of times participants had to switch between the two mental models. Half of the participants began with the Customization Layer block, the other half with the Control condition. Within the Customization Layer block, the presentation order of the three variants was fully counterbalanced. The settings were randomly ordered across all blocks.

The experiment procedure was similar to the MTurk study, except for a few changes reflecting the within-subject design. After each mechanism condition, participants were asked to *rate* on a 7-point Likert scale the mechanism on three metrics: ease of use, perceived speed, and satisfaction (Appendix C.4.2). At the end of the experiment, participants were asked to *rank* the four mechanisms on the same metrics (Appendix C.4.3). No recognition questionnaires were administered, since it would have been difficult to tease apart the learning that happened in each condition. The experiment was concluded by a brief semi-structured interview. Participants were asked about their perception of the four different customization mechanisms, and we gathered insights on how they developed a mental model of Anchored Customization.

5.2.2 Face-to-Face Lab Results

We first present the quantitative results of this second experiment, followed by the qualitative insights we gathered.

Quantitative Results

We ran a repeated measures ANOVA on the duration of trials. As in Experiment 1, the data was log-transformed and we used medians. As shown on Figure 5.4, there was a main effect of customization mechanism ($F_{3,33}$ = 5.61, p < .01, η^2_G =

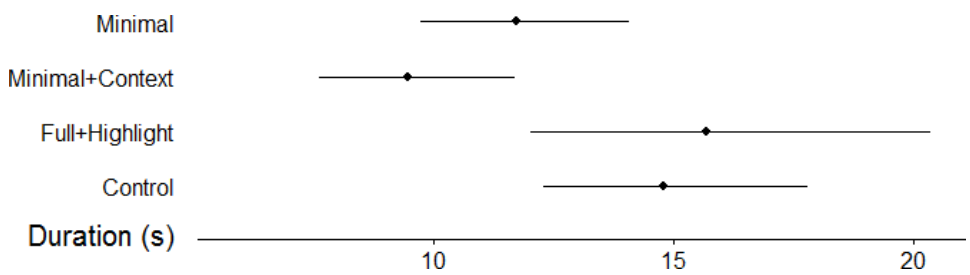


Figure 5.4: Experiment 2: 95% confidence intervals of the median duration per Customization Mechanism. M+C is significantly faster than F+H and Control, but not significantly different from M. (N=12)

0.16). M+C was faster than Control ($p < .05$, 36% faster) and F+H ($p = .06$, 40% faster). No other pair was significantly different. The order in which participants saw the conditions (Control first, or Customization Layer first) had no significant effect either.

The subjective ratings collected after each block were analyzed with a Friedman test. No differences were found for satisfaction and ease of use. Perceived speed was borderline significant ($\chi^2_3 = 7.1$, $p = .07$), with the majority of the difference coming from M+C reported to be faster than F+H ($p = .1$). In terms of the ranking data, participants were almost evenly divided between Customization Layer (6 top ranks) and Control (7) —one participant ranked two mechanisms as best. There was also no clear consensus among which of the three CL variants was best: with 1, 3, and 2 votes going to M, M+C, and F+H respectively.

Qualitative Results

The comments made by participants during the experiment and their answers during the interview were recorded by the experimenter. We report the findings of a thematic analysis [6].

Anchored Customization Nine out of 12 participants acquired the intended mental model of Anchored Customization: 6 acquired it immediately with the first variant of Customization Layer they encountered, the other 3 with the second variant. Once participants understood the concept of Anchored Customization, they were able to use it successfully: *"I'm used to all the settings hidden away in a menu, but I think this (Minimal) makes a lot of sense"* (P5), *"I think the point of this (F+H) is that I don't need to think under which category each setting is"* (P3). In some cases, participants even imagined appropriate anchors that were

not actually present in Wunderlist, which shows a clear grasp of the Anchor Customization Concept. For instance, P1 said: *"I was looking for a printing icon"*, while searching for the setting "don't print completed todos when printing a list".

Three out of 12 participants did not seem to have acquired the intended mental model: in more than 50% of trials, they clicked on any anchor, from which they immediately opened the full panel and browsed the tabs to find the desired setting. P4, who was in the Minimal condition, even complained: *"I don't like 'showing all' all the time, it's too many clicks".* These 3 participants were among the 6 who started the experiment in the Control condition, which could have negatively affected their behavior in the subsequent Customization Layer conditions. The only time these participants searched for an appropriate anchor (rather than just any anchor) was when the target setting was related to one of the Smart Lists, in the left sidebar (see Figure 4.5). In these cases, participants may have reasoned by analogy with their practice trial, which instructed them to change a (different) Smart Lists setting.

Customization Layer Variants Interestingly, at least 3 participants did not notice any difference between M and M+C during the trials, and only realized that they were different at the end when they were asked to rank the four customization mechanisms. Some participants liked the extra structure provided by the tab names in M+C: *"In Minimal I don't know where I am in the greater structure. [M+C] is more organized" (P1)*; *"The links in [M+C] are clearer, you know where you're going"* (P8).

In F+H, at least 4 participants did not see the orange highlighting, although it was clearly visible (Figure 4.7). This could be a case of inattentional blindness [34]: participants who hadn't yet acquired the intended mental model may have discarded the highlighting because it had no meaning to them. Other participants thought of the highlighting as a soft suggestion from the system: *"At one point I assumed the highlighting was like a hint"*(P8), *"[the system] is saying: I think you are looking for this, but I'm not sure"* (P12). However, no participant seemed to notice the *reverse* highlighting, from the settings panel to the anchors. This is a relatively subtle design element, which may not be noticeable in a short experiment with constraint tasks.

A few participants suggested ideas for yet another variant of the Customization Layer. The settings would be displayed in an even more minimal panel, showing only the anchored settings. Instead of providing a "show all" link at the bottom of the panel itself, the full settings panel could be accessed from its usual location, typically via a menu at the top of the screen or a cog icon. Providing a fixed access points to all the settings may match users' mental models better: *"If a setting*

doesn't have an anchor, I would look for a general menu with all settings" (P12). It would also address the concern of P10: *"When clicking on show all, you are not sure that it will show all the settings"* —because the "show all" link appears in an otherwise very contextualized panel.

Wunderlist Although it was not an explicit goal, this experiment revealed some usability problems with Wunderlist's settings panel, which was used as the Control condition. The most problematic aspect is that 10 settings are hidden under a "show more" button in the Shortcuts tab. This design decision was probably intended to avoid showing all 20 shortcuts settings at once, by hiding the ones less likely to be changed. However, this "show more" button was not salient enough, and 10/12 participants missed it the first time they looked for one of the settings it was hiding. Interestingly, this problem was naturally alleviated in all Customization Layer variants: in both M and M+C, only the relevant settings are shown when clicking on an anchor, so there is no need for a hiding mechanism. F+H automatically unhides all of the 10 more advanced shortcuts if one of them is highlighted, and scrolls down to the first highlighted setting.

The categories chosen by Wunderlist's designers seemed to generally make sense for our participants. Yet a few settings appeared to be misclassified, which resulted in longer search times. For instance, the setting "enable sound for new notifications" was in the General tab, next to "enable sound for checking-off a todo". However, most participants looked for it first under the Notifications tab, which is a more precise category. This kind of ambiguity doesn't arise in Anchored Customization, since the same setting can be mapped to any meaningful anchor.

Unsurprisingly, we found that the quality of the text labels plays a key role in helping users find settings. Indeed, participants appear to "scan" each tab very quickly, looking for keywords that would match the instructions they were given. Because of the vocabulary problem [19], finding the appropriate keywords to describe a setting can be challenging for designers. But once a word is chosen, it should be used consistently throughout the settings panel. For instance, our participants were very confused by the fact that notifications are sometimes referred to as "activities" in the current version of Wunderlist.

Visual Design Our primary concern while designing the Customization Layer was interaction design, not aesthetics. However, as is often the case in HCI experiments, some participants tended to focus on the visual appearance of the different interfaces more than their behavior: 4/12 participants indicated that they liked the look of the Wunderlist settings panel better. In particular, the presence of an icon

on each tab was appreciated (P9). This might explain why 7 participants ranked Control as well or above Customization Layer: as P10 puts it, *"for customers pretty is more important"*.

The visual design of our Customization Layer was optimized to remain usable across apps with different visual styles. We chose a dark overlay with white anchors to maximize visibility, independently of the color palette of the underlying application. P5 found it too dark, and not colorful enough. Furthermore, P5 also disliked the fact that the anchors were in some places not well aligned with each others, or that the elements inside them were not properly centered. This is a consequence of generating anchors from regular UI elements, which were not originally designed for that purpose. Of course, for commercial-grade software the appearance of an application's customization layer could be tweaked by the designers to be more aesthetically pleasing.

5.3 Secondary Exploratory Analyses

The fact that we obtained different performance results for M and M+C in our two experiments was surprising and prompted us to probe the data further. M was found to be faster than F+H and Control in the first study, whereas it was M+C that was faster than F+H and Control in the second study. In both cases, the p-value and effect sizes are similar. A closer look shows that these results are not contradictory: M and M+C were never found to be significantly different from each other, nor significantly slower than F+H or Control.

In order to tease this apart, we looked at the logged data with a focus to understand participants' approach to finding the target setting. Of particular interest is whether participants found a correct anchor to click on to access the target setting (what we call "anchor search"), or if they defaulted to clicking on any anchor from which they then browsed the full settings panel ("full panel search"). For each participant, we computed the percentage of trials in which they found a correct anchor, out of the total number of trials in the experiment. The distribution of this metric across all participants is clearly bimodal, both on MTurk and in the lab (Figure 5.5). The position of the "valley" (local minimum) of the two distributions is also similar: 61% on MTurk, 63% in the lab.

This analysis points to a likely hidden variable in our data: the search strategy used by participants. Indeed the three Customization Layer conditions vary along this extra dimension, as shown in Figure 5.6. While these variations might explain the difference between the two minimal variants and F+H, the distributions for M and M+C are too similar to explain the difference in results that we observed between the two experiments.

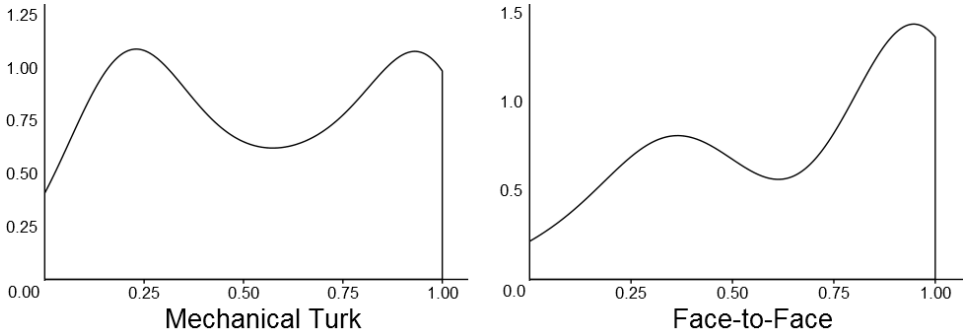


Figure 5.5: Density plots of the distribution of the percentage of trials in which participants selected a "correct" anchor—one associated with the target setting. Participants in the right mode followed an "anchor search" strategy, while participants in the left mode resorted to a "full panel search".

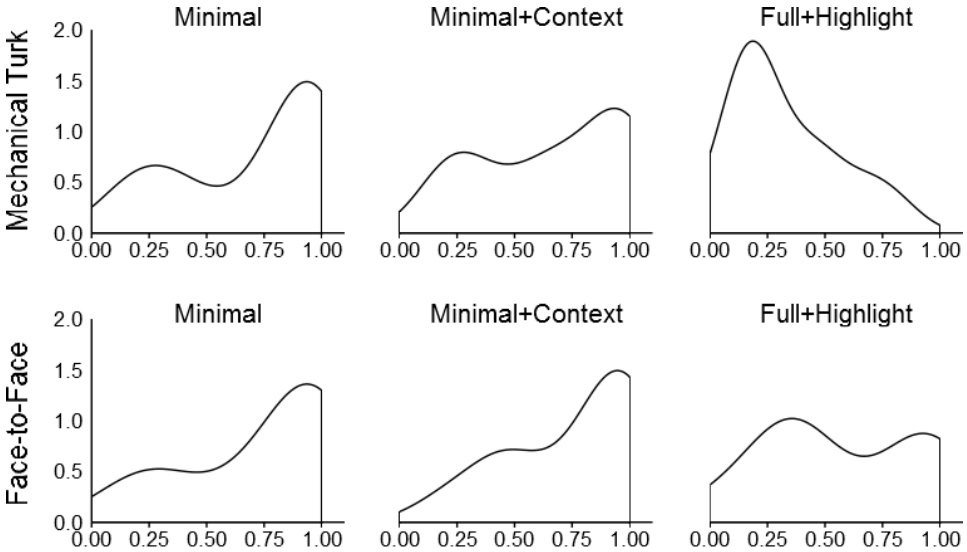


Figure 5.6: Density plots of the percentage of trials in which participants selected a "correct" anchor, for each Customization Layer variant, on MTurk (top) and in the lab (bottom). The minimal variants all have a higher right mode ("anchor search"), while F+H has higher left mode ("full panel search").

For the next step in our exploratory analysis, we looked only at the data from participants who adopted an "anchor search" strategy to see if we could see any differences between M and M+C. More specifically, we re-ran the ANOVA on trial duration only with participants whose average percentage of correct anchor selected was above the local minimum of the bimodal distribution. For the MTurk experiment, M and M+C were both faster than Control ($p < .001$ each), and not significantly different from each other. For the lab experiment, M and M+C were also both faster than both F+H and Control (all $p < .05$). This secondary analysis suggests that the results of MTurk and the face-to-face experiments may be consistent, as long as you take search strategy into account: M and M+C are *both* faster than Control, and not significantly different from each other. As with any non-planned analysis, these results need to be interpreted with caution.

Chapter 6

Discussion

The results of our experiments are promising: whether on MTurk or in the lab, one of the minimal variants was significantly faster than Control, with a medium effect size; and no variant was slower than Control. These performance improvements were obtained even though participants were exposed to Anchored Customization for the first time, and received little information to help them build an appropriate mental model. We now discuss the insights gathered on the three Customization Layer variants, and reflect on the possibilities offered by this new customization mechanism.

6.1 Reconciling the Performance Results

M and M+C performed differently in the two experiments. Our secondary analyses revealed a likely hidden variable, namely search strategy. For participants using the anchor search strategy, the results of the two experiments are consistent: M and M+C are *both* faster than Control, and not significantly different from each other. Hence there must be a significant difference in how the "full panel search" participants performed in the two experiments which translated into different relative performances for M and M+C on MTurk and in the lab. It might just be a statistical fluke. For instance, looking at the data shows that on MTurk all 5 of of the M+C participants that did a "full panel search" were disproportionately slower than the "full panel search" participants in *all* other conditions. This inconsistency suggests that the randomization of participant assignment to conditions may not have equalized individual differences. More research will be needed to further assess this issue.

Few participants in the F+H condition used the "anchor search" strategy: 2/12 on MTurk, 5/12 in the lab. It could be that highlighting settings in a panel is not a strong enough cue to help users acquire the Anchored Customization mental model. Since the structure of the settings panels is retained in F+H, there may be a strong transfer effect that encourages users to default to the "full panel search", instead of exploring anchors. Since few participants used F+H the way it was intended, we cannot conclude with certainty on its potential performance: is F+H necessarily slower than M and M+C, or can it be as fast when properly used? In

any case, F+H is not slower than Control, so it would not be detrimental. Since some participants perceived the highlighting as a hint, F+H could be used as a "softer" version of Customization Layer for users who might not be comfortable with the degree of minimalism of the two minimal variants.

6.2 Performance/Awareness Trade-Off

On MTurk, M was significantly faster than F+H and Control, but it also scored significantly lower on the recognition of tab names. This could be interpreted as a performance/awareness trade-off found in other multi-layered interfaces [15]. Yet, users in M did just as well as others in terms of recognizing the settings themselves. Thus, the awareness trade-off here seems to affect only awareness of the *structure* of the upper layer (the full settings panel), not its content, as there were no differences in recognition scores for the settings themselves. This is not entirely surprising: in Customization Layer, all the settings are accessible via the minimal panel, albeit from different anchors. By contrast, the personalized interfaces studied by Findlater *et al.* [15] only display a static subset of features in the first layer, while the others were only visible in a different layer.

6.3 Applicability to Other Desktop Software

We focused our work on task management applications, and the question remains whether Anchored Customization could provide similar benefits for other types of software. At one extreme is text editors, which are often highly customizable. These editors typically have very few always-visible UI elements, and rely mostly on menus and keyboard shortcuts. Thus, they are not well-suited for Anchored Customization. Furthermore, text editor users are generally comfortable customizing their software by editing the config files directly.

The opposite extreme is complex software applications designed for non-technical users, such as Adobe Creative Suite. These applications have many widgets that provide access to lots of features. Anchoring settings to these various UI elements is possible, but the resulting Customization Layer may be overwhelming. However, the main interface of these applications can itself be overwhelming, especially for new users. Anchored Customization would simply reflect the complexity of the underlying software.

6.4 Applicability to Handheld Devices

Our systematic review showed that mobile apps organize their settings in very similar ways to desktop applications. Mobile apps generally rely on icons and buttons for user input, since keyboard shortcuts and extensive menus are not available. As such, they are well suited for Anchored Customization. The limited screen real estate would warrant using a minimal variant. The anchored customization mechanism could be activated via a standard application menu, but touchscreens offer other possibilities: for instance, users could long-press or multi-tap an anchor to see its associated settings, or use a special standardized gesture to activate the Customization Layer. We observed that mobile applications generally offer fewer settings than desktop apps. The introduction of a well-designed customization mechanism could lead to more customizable mobile apps.

6.5 Interpretation Gulf

Anchored Customization was designed to reduce the gulf of execution inherent with settings panels, but it may also reduce the gulf of *interpretation* [32]. In traditional settings panels, users do not always know if modifying a settings had the intended effect. Anchored Customization mitigates this problem by reducing the spatial and temporal indirection [3] between settings and the user interface, especially for settings that cause an immediate visual change to their anchors. In any case, designers should provide an easy "undo" feature, especially for settings that don't cause an immediate visual change. A simple solution would be to add a backward arrow in the "Preferences" or "Customize" entry in the usual application menu.

6.6 The Developers' Point of View

Beyond its benefits to users, our Anchored Customization approach may also change the way designers and developers think about customization. Although they are not *required* to change any setting to adopt Anchored Customization, the process of mapping settings to UI elements could have a positive effect on the settings offered. For instance, designers might realize that some parts of the interface have no setting anchored to them, which would highlight a potential opportunity for providing settings that cater to this area. Creating and labeling settings may also become faster, since the problem of finding appropriate words to refer to interface elements is mitigated by the context provided by the anchors.

Finding categories to organize settings into tabs might also become superfluous.

6.7 Limitations

While our results are promising, our experiments had limitations, which come mostly from the challenges of evaluating customization mechanisms in an artificial setting.

Personalization systems are indeed notoriously difficult to evaluate. End-user customization occurs infrequently in the real world, and at a varying degree depending on the individuals, the software they use, and a variety of external factors [27]. Long-term field experiments, such as the one conducted by McGrenere *et al.* [30], best capture these behaviors and measure the effects of customization mechanisms. These field experiments are typically long and costly to run, and as a result seldom undertaken. The work presented here did not aim to study complex personalization behaviors over time. Instead, we focused on the usability (or lack thereof) of the customization mechanisms present in today's consumer software. Evaluating the long-term effects of the Anchored Customization approach remains future work.

Furthermore, we compared customization mechanisms mainly on how quickly participants could find a designated setting. However, in a real world situation, the *awareness* (or lack thereof) of which settings are available likely plays an important role. In traditional settings panels, awareness is gained by serendipitous discovery (also referred to as "incidental learning" [15]) : users happen to notice a setting of interest while searching the panel for another one. Serendipity can also happen in Anchored Customization, but the notion of proximity is relative to the anchors, instead of the settings panel's structure. Because our experiment task was time-constrained, these different forms of serendipity were not well captured.

To maximize ecological validity, we used Wunderlist's actual settings panel as a baseline for the Control condition. But some participants focused their feedback on its high quality visual design relative to our prototyped Customization Layer variants. Recreating this panel in the same visual style as our prototypes might have increased the internal validity of our experiments.

In retrospect, the practice trial that participants were given could have been more effective at conveying the intended model. Participants performed only one trial, thus only had to click on one anchor to complete it. However, to really understand the concept of Anchored Customization, one must click on at least two anchors to see that the settings offered are different. Some participants took time during the practice trial to explore the Customization Layer on their own, clicking on multiple anchors to see the outcome. This free exploration seemed

more effective than our practice trial, and would also be more similar to real-world conditions.

Finally, our experiments only included one application with a medium number of settings. It is possible that the number of settings offered by an app affects the relative performance of Customization Layer and settings panels.

Chapter 7

Conclusions and Future Work

Anchored Customization is an approach that places settings *in context* within the application interface, so that users are not required to learn the abstract structure of a settings panel. A systematic review of a set of Personal Task Management apps found that approximately 84% of the settings could be anchored in the UI. Our Customization Layer prototype reveals all the anchors as affordances for customization. We designed three variants of Customization Layer based on multi-layered interfaces, and implemented these variants on top of a popular web application for task management, Wunderlist. Two experiments (Mechanical Turk and face-to-face) showed that the two minimalist variants were 35-36% faster than Wunderlist's settings panel. A majority of users acquired the intended mental model of Anchored Customization, and were able to use it successfully for finding a variety of settings in Wunderlist. Some users did not seem to acquire the intended mental model, yet they were able to complete the experiment tasks in the familiar structure of a settings panel, thanks to the fallback access we provided in each Customization Layer variant. Better instructions and more guidance might be needed to help these users understand the Anchored Customization approach.

7.1 Long-Term Effects

Evaluating the long term effects of this customization approach remains future work. A longitudinal field study would determine if a more usable customization mechanism actually increases users' likelihood to customize. It could also address subtle, but important questions, such as: are different variants of Customization Layer better suited to users with different level of expertise? Given the opportunity to use multiple variants, as well as a direct right-click access to any anchor's settings, which one would users choose? Would it depend on the users' familiarity with the application and its settings, or is it simply a matter of personal preference?

Our prototype could be distributed to real users of Wunderlist as a browser extension, or adapted to other applications to compare the effect of different types of settings and different application domains. Augmenting multiple applications with our Customization Layer would also help to verify our assumption that An-

chored Customization requires only one-time learning: once this approach is understood in the context of one particular app, the mental model should be transferable to other apps.

7.2 Generating the Mapping

Currently, the designers of an app need to provide the mapping between settings and UI elements. With the growing popularity of advanced front-end frameworks in web development, it might be possible to use code analysis techniques to generate part of the mapping automatically. The interface could be analyzed to determine which elements are affected by a setting (e.g. changing their visibility, position, or appearance), and which elements trigger a function that is itself modified by a setting (e.g. changing the effect of clicking a button). The UI elements identified would automatically become anchors for the appropriate settings. Designers then only need to verify and extend this mapping, anchoring extra settings by exploiting *mental* associations—to complement the *functional* associations that were detected programmatically.

Another possibility would be to involve users in the mapping process. Contrary to crowdsourced contextual help [8], users cannot be expected to *generate* the entire mapping themselves, but they could *tweak* a designer's mapping to better match their expectations. The idea of refining the mapping by aggregating data from individual users could be expanded to other customization opportunities as well. For instance, the most popular extensions and plugins could be anchored to the UI.

7.3 Concluding Remarks

End-user customization is fundamentally a difficult trade-off between upfront costs and uncertain benefits in the future. Users weigh the time and effort needed to customize against the satisfaction of using an interface matching their own taste and preferences, and the performance gains of using a software adapted to their work patterns. For developers and designers, providing multiple versions of an application can require significant additional development costs, which can be difficult to justify in an agile world where the time-to-market is critical. And these efforts might eventually be in vain if users never change the default settings.

Settings panels represent an equilibrium in this trade-off space. Their generic UIs are inexpensive to develop, but their lack of appeal and numerous usability issues can repel users. As a result, few users customize their applications, or do so only rarely. This in turn does not encourage developers to provide better cus-

tomization mechanisms, and so on. Introducing a new customization mechanism therefore requires careful consideration of its costs and benefits.

Much of the prior work reported in the literature has focused on increasing the *potential benefits* of customization, by proposing more powerful or more expressive customization mechanisms. But these expected benefits do not necessarily justify the additional costs incurred by users or developers. By contrast, Anchored Customization minimizes the extra costs for developers by leveraging the customization opportunities already available in existing settings panels. It also reduces the cost of customization for end-users, who can find desired settings faster without having to learn the abstract structure of a settings panel. As such, this novel approach to customization has the potential to be adopted widely. Our ultimate goal is to increase users' satisfaction by interacting with applications better suited to their own needs and preferences.

Bibliography

[1] Roger Bakeman. Recommended effect size statistics for repeated measures designs. *Behavior Research Methods*, 37(3):379–384, 2005.

[2] Patrick Baudisch, Desney Tan, Maxime Collomb, Dan Robbins, Ken Hinckley, Maneesh Agrawala, Shengdong Zhao, and Gonzalo Ramos. Phosphor: Explaining transitions in the user interface using afterglow effects. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 169–178, New York, NY, USA, 2006. ACM.

[3] Michel Beaudouin-Lafon. Instrumental interaction: An interaction model for designing post-WIMP user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '00, pages 446–453, New York, NY, USA, 2000. ACM.

[4] Ofer Bergman, Ruth Beyth-Marom, Rafi Nachmias, Noa Gradovitch, and Steve Whittaker. Improved search engines and navigation preference in personal information management. *ACM Transactions on Information Systems*, 26(4):20:1–20:24, October 2008.

[5] Anastasia Bezerianos, Pierre Dragicevic, and Ravin Balakrishnan. Mnemonic rendering: An image-based approach for exposing hidden changes in dynamic displays. In *Proceedings of the 19th Annual ACM Symposium on User Interface Software and Technology*, UIST '06, pages 159–168, New York, NY, USA, 2006. ACM.

[6] Virginia Braun and Victoria Clarke. Using thematic analysis in psychology. *Qualitative Research in Psychology*, 3(2):77–101, 2006.

[7] Andrea Bunt, Cristina Conati, and Joanna McGrenere. Supporting interface customization using a mixed-initiative approach. In *Proceedings of the 12th International Conference on Intelligent User Interfaces*, IUI '07, pages 92–101, New York, NY, USA, 2007. ACM.

[8] Parmit K. Chilana, Andrew J. Ko, and Jacob O. Wobbrock. LemonAid: Selection-based crowdsourced contextual help for web applications. In *Pro-*

ceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '12, pages 1549–1558, New York, NY, USA, 2012. ACM.

[9] Parmit K. Chilana, Andrew J. Ko, Jacob O. Wobbrock, and Tovi Grossman. A multi-site field study of crowdsourced contextual help: Usage and perspectives of end users and software teams. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 217–226, New York, NY, USA, 2013. ACM.

[10] Matjaz Debevc, Beth Meyer, Dali Donlagic, and Rajko Svecko. Design and evaluation of an adaptive icon toolbar. *User Modeling and User-Adapted Interaction*, 6(1):1–21, 1996.

[11] Morgan Dixon and James Fogarty. Prefab: Implementing advanced behaviors using pixel-based reverse engineering of interface structure. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 1525–1534, New York, NY, USA, 2010. ACM.

[12] James R. Eagan, Michel Beaudouin-Lafon, and Wendy E. Mackay. Cracking the cocoa nut: User interface programming at runtime. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology*, UIST '11, pages 225–234, New York, NY, USA, 2011. ACM.

[13] W. Keith Edwards, Scott E. Hudson, Joshua Marinacci, Roy Rodenstein, Thomas Rodriguez, and Ian Smith. Systematic output modification in a 2D user interface toolkit. In *Proceedings of the 10th Annual ACM Symposium on User Interface Software and Technology*, UIST '97, pages 151–158, New York, NY, USA, 1997. ACM.

[14] Leah Findlater and Joanna McGrenere. Impact of screen size on performance, awareness, and user satisfaction with adaptive graphical user interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 1247–1256, New York, NY, USA, 2008. ACM.

[15] Leah Findlater and Joanna McGrenere. Beyond performance: Feature awareness in personalized interfaces. *International Journal of Human-Computer Studies*, 68(3):121 – 137, 2010.

[16] Gerhard Fischer. *Adaptive User Interfaces*, chapter Shared Knowledge in Cooperative Problem-Solving Systems – Integrating Adaptive and Adaptable Components, pages 49–68. Elsevier Science Publishers B.V., 1993.

[17] Gerhard Fischer and Elisa Giaccardi. Meta-design: A framework for the future of end-user development. In Henry Lieberman, Fabio Paternò, and Volker Wulf, editors, *End User Development*, volume 9 of *Human-Computer Interaction Series*, pages 427–457. Springer Netherlands, 2006.

[18] Stephen Fitchett, Andy Cockburn, and Carl Gutwin. Improving navigation-based file retrieval. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 2329–2338, New York, NY, USA, 2013. ACM.

[19] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, November 1987.

[20] 6 Wunderkinder GmbH. Wunderlist. Version 3.12.3, 2015. http://www.wunderlist.com (Retrieved on 2015-07-01).

[21] Google. Angular.js. Version 1.3.11, 2015. http://angularjs.org/ (Retrieved on 2015-01-26).

[22] Mona Haraty, Diane Tam, Shathel Haddad, Joanna McGrenere, and Charlotte Tang. Individual differences in personal task management: A field study in an academic setting. In *Proceedings of Graphics Interface 2012*, GI '12, pages 35–44, Toronto, Ont., Canada, Canada, 2012. Canadian Information Processing Society.

[23] Eric Horvitz, Jack Breese, David Heckerman, David Hovel, and Koos Rommelse. The Lumière project: Bayesian user modeling for inferring the goals and needs of software users. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 256–265, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[24] J. Johnson, T.L. Roberts, W. Verplank, D.C. Smith, C.H. Irby, M. Beard, and K. Mackey. The Xerox Star: a retrospective. *Computer*, 22(9):11–26, Sept 1989.

[25] Helge Kahler, Anders Mørch, Oliver Stiemerling, and Volker Wulf. Tailorable systems and cooperative work. *Computer Supported Cooperative Work (CSCW)*, 9(1):1–4, 2000.

[26] Henry Lieberman, Fabio Paternò, Markus Klann, and Volker Wulf. End-user development: An emerging paradigm. In Henry Lieberman, Fabio Paternò, and Volker Wulf, editors, *End User Development*, volume 9 of *Human-Computer Interaction Series*, pages 1–8. Springer Netherlands, 2006.

[27] Wendy E. Mackay. Triggers and barriers to customizing software. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pages 153–160, New York, NY, USA, 1991. ACM.

[28] Thomas W. Malone, Kum-Yew Lai, and Christopher Fry. Experiments with Oval: A radically tailorable tool for cooperative work. *ACM Transactions on Information Systems*, 13(2):177–205, April 1995.

[29] Justin Matejka, Tovi Grossman, and George Fitzmaurice. Patina: Dynamic heatmaps for visualizing application usage. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '13, pages 3227–3236, New York, NY, USA, 2013. ACM.

[30] Joanna McGrenere, Ronald M. Baecker, and Kellogg S. Booth. An evaluation of a multiple interface design solution for bloated software. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, pages 164–170, New York, NY, USA, 2002. ACM.

[31] Xiaojun Meng, Shengdong Zhao, Yongfeng Huang, Zhongyuan Zhang, James Eagan, and Ramanathan Subramanian. WADE: Simplified GUI add-on development for third-party software. In *Proceedings of the 32nd Annual ACM Conference on Human Factors in Computing Systems*, CHI '14, pages 2221–2230, New York, NY, USA, 2014. ACM.

[32] Donald A. Norman. Interfacing thought: Cognitive aspects of human-computer interaction. chapter Cognitive Engineering—Cognitive Science, pages 325–336. MIT Press, Cambridge, MA, USA, 1987.

[33] Ben Shneiderman. Promoting universal usability with multi-layer interface design. In *Proceedings of the 2003 Conference on Universal Usability*, CUU '03, pages 1–8, New York, NY, USA, 2003. ACM.

[34] Daniel J Simons and Christopher F Chabris. Gorillas in our midst: Sustained inattentional blindness for dynamic events. *Perception-London*, 28(9):1059–1074, 1999.

[35] Gunnar Stevens and Torben Wiedenhöfer. Chic - a pluggable solution for community help in context. In *Proceedings of the 4th Nordic Conference on Human-computer Interaction: Changing Roles*, NordiCHI '06, pages 212–221, New York, NY, USA, 2006. ACM.

[36] Wolfgang Stuerzlinger, Olivier Chapuis, Dusty Phillips, and Nicolas Roussel. User interface façades: Towards fully adaptable user interfaces. In *Pro-*

ceedings of the 19th Annual ACM Symposium on User Interface Software and Technology, UIST '06, pages 309–318, New York, NY, USA, 2006. ACM.

[37] Gaurav Paruthi Tao Dong, Mark S. Ackerman, Mark W. Newman. Social Overlays: Collectively Making Websites More Usable. volume 8120 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[38] Volker Wulf and Björn Golombek. Direct activation: A concept to encourage tailoring activities. *Behaviour & Information Technology*, 20(4):249–263, 2001.

Appendix A

Mapping of settings to UI elements in Wunderlist

This appendix shows the JSON file that was used to encode the mapping of settings to UI elements in Wunderlist's web application. UI elements are represented by a CSS selector, which usually consists of CSS classes (starting with a ".") and ids (starting with a "#"). A comma can be used to concatenate multiple CSS selectors into one. The settings associated with each anchor are specified via an array of settings names, which were chosen by the developers of Wunderlist.

```
[{
   "selector": ".addTask-input",
   "settings": ["new_task_location", "shortcut_add_new_task"]
},
{
   "selector": ".starred-wrapper, .star-wrapper",
   "settings": ["behavior_star_tasks_to_top", "
       shortcut_mark_task_starred"]
},
{
   "selector": ".taskItem-duedate, .detail-date .token_0",
   "settings": ["date_format", "time_format", "start_of_week"]
},
{
   "selector": ".detail-checkbox.checkBox, .taskItem-
       checkboxWrapper.checkBox",
   "settings": ["sound_checkoff_enabled", "shortcut_mark_task_done
       "]
},
{
   "selector": ".taskItem-titleWrapper",
   "settings": ["show_subtask_progress", "
       shortcut_select_all_tasks", "shortcut_copy_tasks", "
       shortcut_paste_tasks"]
},
{
   "selector": ".filters-collection .sidebarItem[rel='inbox']",
   "settings": ["shortcut_goto_inbox"]
},
```

```
{
    "selector": ".filters-collection .sidebarItem[rel='all']",
    "settings": ["smartlist_visibility_all", "shortcut_goto_filter_all"]
},
{
    "selector": ".filters-collection .sidebarItem[rel='assigned']",
    "settings": ["smartlist_visibility_assigned_to_me", "shortcut_goto_filter_assigned"]
},
{
    "selector": ".filters-collection .sidebarItem[rel='completed']",
    "settings": ["smartlist_visibility_done", "shortcut_goto_filter_completed"]
},
{
    "selector": ".filters-collection .sidebarItem[rel='starred']",
    "settings": ["smartlist_visibility_starred", "shortcut_goto_filter_starred"]
},
{
    "selector": ".filters-collection .sidebarItem[rel='today']",
    "settings": ["smartlist_visibility_today", "shortcut_goto_filter_today", "today_smart_list_visible_tasks"]
},
{
    "selector": ".filters-collection .sidebarItem[rel='week']",
    "settings": ["smartlist_visibility_week", "shortcut_goto_filter_week", "today_smart_list_visible_tasks"]
},
{
    "selector": "#user-toolbar .activities-count, .detail-reminder .wundercon.reminder",
    "settings": ["notifications_desktop_enabled", "notifications_email_enabled",
       "notifications_push_enabled", "sound_notification_enabled", "shortcut_show_notifications"
    ]
},
{
    "selector": ".user-avatar, .user-name, .user-arrow",
    "settings": ["shortcut_sync", "shortcut_goto_preferences"]
},
{
    "selector": "#search-toolbar .search-icon",
    "settings": ["shortcut_goto_search"]
```

```
},
{
   "selector": ".sidebarActions-addList",
   "settings": ["shortcut_add_new_list"]
},
{
   "selector": ".actionBar-bottom a.tab.share",
   "settings": ["shortcut_send_via_email"]
},
{
   "selector": ".completed-items-heading",
   "settings": ["print_completed_items"]
},
{
   "selector": ".actionBar-bottom a.tab.last-tab",
   "settings": ["print_completed_items", "shortcut_copy_tasks", "
       shortcut_paste_tasks"]
},
{
   "selector": ".detail-trash",
   "settings": ["confirm_delete_entity", "shortcut_delete"]
}]
```

Appendix B

Experiment 1 Resources

This appendix contains resources used in the *remote Mechanical Turk experiment*, discussed in Chapter 5.1.

B.1 Tasks

We provide here the instructions given to participants for each of the 40 tasks. Each task has two formulations, depending on the default value of the setting. The first formulation corresponds to changing the setting *from* the default value provided by Wunderlist; the second corresponds to changing the setting *back to* its default value, starting from the *opposite* of this default value.

- Change the date format to DD.MM.YYYY
  Change the date format to MM/DD/YYYY
- Change the time format to 12 hour
  Change the time format to 24 hour
- Change the first day of the week to be Sunday
  Change the first day of the week to be Monday
- Enable sound when checking-off a todo
  Disable sound when checking-off a todo
- Enable sound for new notifications
  Disable sound for new notifications
- When creating a new todo, add it at the top of the current list
  When creating a new todo, add it at the bottom of the current list
- When deleting a to-do, ask me for confirmation
  When deleting a to-do, don't ask me for confirmation
- When starring a to-do, move it to the top of the list
  Do not automatically move starred to-dos to the top of the list

- When printing a todo list, do not print the completed to-dos
  When printing a todo list, print also the completed to-dos
- Show subtask progress on to-dos
  Hide subtask progress on to-dos
- Change shortcut for adding a new todo to CTRL + 0
  Change shortcut for adding a new todo to SHIFT + N
- Change shortcut for creating a new list to CTRL + L
  Change shortcut for creating a new list to CTRL + K
- Change shortcut for checking off todos to CTRL + D
  Change shortcut for checking off todos to CTRL + H
- Change shortcut for starring todos to CTRL + S
  Change shortcut for starring todos to CTRL + M
- Change shortcut for selecting all todos to CTRL + A
  Change shortcut for selecting all todos to CTRL + Q
- Change shortcut for deleting a list or a todo to CTRL + BACKSPACE
  Change shortcut for deleting a list or a todo to SHIFT + BACKSPACE
- Change shortcut for copying todos to CTRL + C
  Change shortcut for copying todos to SHIFT + M
- Change shortcut for pasting todos to CTRL + V
  Change shortcut for pasting todos to SHIFT + P
- Change shortcut for selecting the search box to CTRL + F
  Change shortcut for selecting the search box to CTRL + G
- Change shortcut for opening preferences to CTRL + P
  Change shortcut for opening preferences to CTRL + .
- Change shortcut for sharing a list via email to CTRL + E
  Change shortcut for sharing a list via email to CTRL + U
- Change shortcut for showing the activities panel to CTRL + SHIFT + A
  Change shortcut for showing the activities panel to CTRL + SHIFT + D
- Change shortcut for going to inbox to CTRL + I
  Change shortcut for going to inbox to SHIFT + B

- Change shortcut for showing the list of the todos assigned to you to CTRL + 1
  Change shortcut for showing the list of the todos assigned to you to SHIFT + A
- Change shortcut for showing the list of the todos that are starred to CTRL + 2
  Change shortcut for showing the list of the todos that are starred to SHIFT + S
- Change shortcut for showing the list of the todos due today to CTRL + 3
  Change shortcut for showing the list of the todos due today to SHIFT + T
- Change shortcut for showing the list of the todos due this week to CTRL + 4
  Change shortcut for showing the list of the todos due this week to SHIFT + W
- Change shortcut for showing the list of all the todos to CTRL + 5
  Change shortcut for showing the list of all the todos to SHIFT + L
- Change shortcut for showing the list of the completed todos to CTRL + 6
  Change shortcut for showing the list of the completed todos to SHIFT + C
- Change shortcut for syncing with the server to R
  Change shortcut for syncing with the server to S
- In the left sidebar, show the list of todos assigned to you
  In the left sidebar, hide the list of todos assigned to you
- In the left sidebar, show the list of todos that are due today
  In the left sidebar, hide the list of todos that are due today
- In the left sidebar, hide the list of todos that are due this week
  In the left sidebar, show the list of todos that are due this week
- In the left sidebar, hide the list of all the todos
  In the left sidebar, show the list of all the todos
- In the left sidebar, show the list of todos that are already completed
  In the left sidebar, hide the list of todos that are already completed
- In the lists of tasks due today and this week, show all to-dos
  In the lists of tasks due today and this week, show only the todos assigned to me

- Enable email notifications
  Disable email notifications
- Enable push notifications
  Disable push notifications
- Enable desktop notifications [duplicated]
  Disable desktop notifications [duplicated]

B.2 Participant Consent Form

THE UNIVERSITY OF BRITISH COLUMBIA

Department of Computer Science
2366 Main Mall
Vancouver, B.C., V6T 1Z4

Consent Form

Principal Investigator

Dr. Joanna McGenere, Associate Professor, Department of Computer Science, University of British Columbia (###) ###-####—######@#####.

Co-Investigator

Antoine Ponsard, MSc student, Department of Computer Science, University of British Columbia (###) ###-####—######@#####.
Summaries of the data and anonymous samples of comments obtained in this study may be included in a doctoral dissertation and in publications that arise from the research.

Study Purpose

The goal of this study is to evaluate the usability of a new interface for customizing software applications, which allows users to change pre-defined settings. In particular, we are interested in comparing this new interface with the traditional settings panel found in many software applications.

Study Procedures

This study is expected to require less than 45 minutes of your time. First, you will be asked to create a temporary account on Wunderlist.com, a popular todo list application, and to setup the experiment software by adding a bookmark to your web browser. Then you will be given a brief tutorial of how to use Wunderlist, and a set of tasks to complete using either Wunderlist's default settings panel or our new interface. You will be asked to fill out a questionnaire about the application, as well as some demographics information. Finally you will be asked to interact informally with a version of our prototype designed to operate with Gmail, and we will you your feedback on the prototype. The interview will be audio recorded.

Compensation

We are very grateful for your participation in this study. You will receive a small honorarium ($10) for your participation

Potential Risks

There are no known risks to your participation in this study.

Confidentiality
Only the principal investigator and co-investigators will have access to the data and the audio recordings of the interviews (all data will be stored in a password-protected location). Confidentiality will be maintained by anonymizing the identity of participants.

Contact for information about the study
Please contact Antoine Ponsard (email: #####@#####) if you need more information about the study.

Contact for information about the rights of research subjects

If you have any concerns or complaints about your rights as a research participant and/or your experiences while participating in this study, contact the Research Participant Complaint Line in the UBC Office of Research Services at ###-###-#### or if long distance e-mail #####@##### or call toll free #-###-###-#### (Toll Free: #-###-###-####).

Consent:
My participation is entirely voluntary and I may refuse to participate or withdraw from the study at any time.

My signature below indicates that I have received a copy of this consent form for my own records.

My signature indicates that I consent to participate in this study.

_____

Participant Signature Date

_____

Printed Name of the Participant signing above

B.3 Questionnaires

B.3.1 Demographics

Mechanical Turk participants were administered an electronic version of the demographics questionnaire used Experiment 2 (Appendix C.4.1).

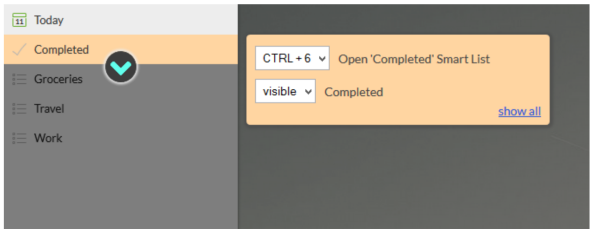
B.3.2 Satisfaction

At the end of the experiment, participants were asked to rate the mechanism they were using on three metrics: ease of use, perceived speed, and satisfaction.

B.3.3 Recognition

At the end of the experiment, participants completed two recognition questionnaires: one on tab names, the other on individual settings. In both cases, half of the answers were made up by the authors, but plausible.

Here is the customization interface that you've used in this experiment:



In this customization interface, how easy or hard was it to find the settings you were looking for?

difficult	-3	-2	-1	0	1	2	3	easy

Please rate how you liked or disliked this customization interface:

didn't like it	-3	-2	-1	0	1	2	3	liked it

Please share your feedback with us on this customization interface: What did you like about it? What could be improved?

If you encountered a bug in the experiment software (= something was not working as it should have), please let us know here:

The settings menu you used was organized into tabs. Please select the tabs that you remember in the list below:
(note: some of these do not appear in Wunderlist! You will receive an extra **$0.03** for each correct response, minus **$0.03** for each incorrect response, but not less than $0.00.)

Display	Yes	No
Smart Lists	Yes	No
Account	Yes	No
Security	Yes	No
Reminders	Yes	No
General	Yes	No
Sounds	Yes	No
Notifications	Yes	No
Sync	Yes	No
Shortcuts	Yes	No

Submit

Among the settings listed below, select the ones that you remember seeing in Wunderlist:

The text in [brackets] indicate a possible value of each setting, just to help you remember. Note: some of these settings do not appear in Wunderlist!

You will receive an extra **$0.03** for each correct response, minus **$0.03** for each incorrect response, but not less than $0.00.)

Open 'Completed' Smart List [CTRL + 6]	◯ Yes	◯ No
Show details panel when adding an item	◯ Yes	◯ No
Tomorrow [visible]	◯ Yes	◯ No
Open 'Assigned to Me' Smart List [CTRL + 1]	◯ Yes	◯ No
SMS Notifications	◯ Yes	◯ No
Open Sharing Preferences [CTRL + S]	◯ Yes	◯ No
Week & Today Settings [Show all items]	◯ Yes	◯ No
Date Format [DD.MM.YYYY]	◯ Yes	◯ No
Confirm before deleting list	◯ Yes	◯ No
Completed [visible]	◯ Yes	◯ No
Paste tasks [CTRL + V]	◯ Yes	◯ No
Edit Item Details [CTRL + E]	◯ Yes	◯ No
Open 'Upcoming' Smart List [CTRL + 7]	◯ Yes	◯ No
Print Selected List [CTRL + P]	◯ Yes	◯ No
Sync [R]	◯ Yes	◯ No
Time Zone [EST (GMT-5)]	◯ Yes	◯ No
Overdue [visible]	◯ Yes	◯ No
Copy tasks [CTRL + C]	◯ Yes	◯ No
Time Format [12 hour]	◯ Yes	◯ No
Open 'Starred' Smart List [CTRL + 2]	◯ Yes	◯ No

Submit

Appendix C

Experiment 2 Resources

This appendix contains resources used in the *face-to-face lab experiment*, discussed in Chapter 5.2.

C.1 Tasks

The same tasks were used than in Experiment 1 (Appendix B.1), but only the first formulation—corresponding to Wunderlist's default values.

C.2 Call For Participation

This recruitment poster was posted in multiple locations on the UBC campus.

$10 for testing new customization interfaces

We are looking for participants for a study of a new interface for customizing software applications, in replacement of the traditional settings panel. You will receive a compensation of $10 for your participation in a single 45 minutes session.

when: Monday August 24 – Thursday August 27

where: ICICS building (Computer Science)

duration: 45 minutes

To participate in the study, please email Antoine Ponsard (######@######) to schedule a time. Thanks!

Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######
Antoine Ponsard ######@######

C.3 Participant Consent Form

THE UNIVERSITY OF BRITISH COLUMBIA

Department of Computer Science
2366 Main Mall
Vancouver, B.C., V6T 1Z4

Consent Form

Principal Investigator

Dr. Joanna McGenere, Associate Professor, Department of Computer Science, University of British Columbia (###) ###-####—#####@#####.

Co-Investigator

Antoine Ponsard, MSc student, Department of Computer Science, University of British Columbia (###) ###-####—#####@#####.
Summaries of the data and anonymous samples of comments obtained in this study may be included in a doctoral dissertation and in publications that arise from the research.

Study Purpose

The goal of this study is to evaluate the usability of a new interface for customizing software applications, which allows users to change pre-defined settings. In particular, we are interested in comparing this new interface with the traditional settings panel found in many software applications.

Study Procedures

This study is expected to require about 30 minutes of your time. First, you will be asked to create a temporary account on Wunderlist.com, a popular todo list application, and to setup the experiment software by adding a bookmark to your web browser. Then you will be given a brief tutorial on how to use Wunderlist, and a set of tasks to complete using either Wunderlist's default settings panel or our new interface. Finally you will be asked to fill out a questionnaire about the application, as well as some demographic information.

Compensation

We are very grateful for your participation in this study. You will receive a small honorarium ($2.00) for your participation, plus a bonus of up to $2.90 depending on your performance.

Potential Risks

There are no known risks to your participation in this study.

Confidentiality
Only the principal investigator and co-investigators will have access to the data. Confidentiality will be maintained by anonymizing the identity of participants.

Contact for information about the study
Please contact Antoine Ponsard (email: #####@#####) if you need more information about the study.

Contact for information about the rights of research subjects

If you have any concerns or complaints about your rights as a research participant and/or your experiences while participating in this study, contact the Research Participant Complaint Line in the UBC Office of Research Services at ###-###-#### or if long distance e-mail #####@##### or call toll free #-###-###-#### (Toll Free: #-###-###-####).

Consent:
My participation is entirely voluntary and I may refuse to participate or withdraw from the study at any time.

My signature below indicates that I have received a copy of this consent form for my own records.

My signature indicates that I consent to participate in this study.

_____

Participant Signature Date

_____

Printed Name of the Participant signing above

C.4 Questionnaires

C.4.1 Demographics

After signing the consent form, participants were asked to complete a short demographics questionnaire.

P___

Age:

☐ Prefer not to say

Gender:

- ○ Male
- ○ Female
- ○ Other
- ○ Prefer not to say

How often do you use a computer, approximately?

- ○ daily, more than 4 hours
- ○ daily, less than 4 hours
- ○ a few times a week
- ○ rarely

Have you ever used Wunderlist?

- ○ never
- ○ I tried it before, but don't use it regularly
- ○ regular user, less than a year
- ○ regular user, more than a year

Do you regularly use:

- ☐ the Wunderlist smartphone app?
- ☐ the Wunderlist web or desktop app?

C.4.2 Ratings

At the end of each condition, participants were asked to rate the mechanism they were using on three metrics: ease of use, perceived speed, and satisfaction.

C.4.3 Rankings

This questionnaire was administered at the end of the experiment, after all four blocks of 10 trials were completed.

P___

Please rank the four customization interfaces (A, B, C, D) on the following criteria:

Which one made it the **easiest** to find the settings you were looking for? *(1 is easiest)*

Interface A	1	2	3	4
Interface B	1	2	3	4
Interface C	1	2	3	4
Interface D	1	2	3	4

Which one was the **fastest** to find the settings you were looking for? *(1 is fastest)*

Interface A	1	2	3	4
Interface B	1	2	3	4
Interface C	1	2	3	4
Interface D	1	2	3	4

Which one did you **like** the best? *(1 is most liked)*

Interface A	1	2	3	4
Interface B	1	2	3	4
Interface C	1	2	3	4
Interface D	1	2	3	4

Well done!

In this customization interface, how easy or hard was it to find the settings you were looking for?

difficult	○	○	○	○	○	○	○	**easy**
	-3	*-2*	*-1*	*0*	*1*	*2*	*3*	

How quickly or slowly were you able to find the settings you were looking for?

slow	○	○	○	○	○	○	○	**fast**
	-3	*-2*	*-1*	*0*	*1*	*2*	*3*	

Please rate how you liked or disliked this customization interface:

didn't like it	○	○	○	○	○	○	○	**liked it**
	-3	*-2*	*-1*	*0*	*1*	*2*	*3*	