Web Design Patterns for Mobile Devices

JORGE RIBEIRO, Faculdade de Engenharia, Universidade do Porto
MIGUEL CARVALHAIS, ID+ / Faculdade de Belas Artes, Universidade do Porto

Mobile devices have brought new constraints and possibilities to the field of web design. You can no longer design a website without thinking how it will work on a mobile device: the whole mobile experience needs to be designed from the beginning. Because designing for these devices is considerably different from designing for wider screens, you cannot only rely on your old techniques: you will need new ones. In this paper we propose a set of twenty-one patterns for designing web interfaces on mobile devices.

1. INTRODUCTION

The patterns proposed on this paper began from the interest to further explore the design of interfaces, particularly web interfaces, on mobile devices. We currently have at our disposal a new, universe of portable devices that follow us wherever we go, are always on, and have continuous and easy access to the web. We also have gestural interfaces that provide for more natural and appealing interactions. Because these devices are relatively new, there are not strong standards or guidelines to help us to design for them. Moreover, because of their novelty, users do not have so rigid expectations of how an interface should be, as they may have for desktop environments. We think that this places us in a favorable position to explore new types of interactions, and to reinforce the importance and need to research new ways to improve the design of dedicated interfaces for mobile devices.

While in this work we talk about designing for mobile in general, we decided that it would be more worthwhile to focus our efforts on the design of interfaces for higher-end devices with touch interfaces, i.e., devices that are colloquially named as smartphones. Nonetheless, part of the results of this work can still be useful for the design of website for less capable devices, such as features phones.

With this in mind we decided to approach this problem through a pattern based methodology. The patterns described in this work aspire to help us to design better and more satisfying experiences on the mobile web.

2. DESIGNING FOR MOBILE DEVICES

In the last few years, the growth of mobile devices with more capable browsers, and permanent access to the web reinforced the need to design websites optimized for that class of devices, but like any new medium they impose new constraints and capabilities that should be considered.

A noticeable difference between desktop and mobile environments, and probably the most significant constraint that one faces when designing or adapting a website for a mobile view, is the screen size: mobile devices are considerably smaller than their desktop counterparts. On the desktop, 98%[1] of users browse the web on displays with resolutions higher than 1024x768 pixels, while on mobile, screen resolutions are much lower. According to StatCounter, on the first trimester of 2012 the most common screen resolution on mobile devices was 320x480 pixels with 20,62%,[2] but these values vary significantly with the type of device. If we take as reference the 1024x768 pixel resolution and compare it to the most common resolution of mobile devices, we lose around 80% of the screen real estate. Therefore, with less available space, websites need to be rethought to address the space limitations.

A related problem is the variability in screen size, resolutions and proportions that are available in the current mobile device landscape. The 320x480 pixels resolution is still the most common but we are faced with a myriad of devices with a diverse range of screen sizes and resolutions that support the need to design interfaces that require designing interfaces with unbound dimensions. We cannot foresee which devices will access our websites, so a design strategy base on a responsive design, in which we design a fluid layout that

[1] Based on the data collected from W3schools (http://w3schools.com) from January 2012. It only reports to accesses to their website, so it might not be representative of other websites.
[2] According to the data from StatCounter (http://gs.statcounter.com/#mobile_resolution-ww-monthly-201201-201204-bar), accessed on 1 June 2012

adapts gracefully to whatever screen size, is a better approach than design for each individual resolution with fixed widths.

Another significant difference between desktop and mobile interfaces is the way in which we interact with them. Traditional desktop environments rely primarily on mouse-driven interactions, in which an artificial pointing device acts as an intermediary between our movements and what happens on screen, whilst smartphones have touch interfaces, in which there is a more direct manipulation of the elements on the screen.

There are also human constraints that drive the design of mobile interfaces. The human finger is a much more imprecise pointer than a mouse, so it is not possible to accurately hit targets much smaller than the size of the fingertip. This is even more problematic for people with less dexterity, particularly older people, and for the "rushed and distracted user" (Clark 2010, 13). Therefore, targets on touch interfaces need to be properly sized and positioned (Wroblewski 2011, 67). Moreover, the lack of haptic feedback on current touch screens does not help with this problem. Touch interfaces also have their advantages, besides the more natural and pleasing interactions, it is possible to explore the use gestures, — e.g., the pinch to zoom —, as a way to improve the user experience.

The mouse-over technique is a common pattern on the web that is used for revealing additional information, e.g., tooltips, but it does not work on touch devices because there is no cursor to hover on page elements. Therefore, interactions that rely merely on mouse-over need to be rethought (Wroblewski 2011, 78), so no critical information is left behind hover-based interactions.

Since we are talking about designing for the web, an important factor is the browser itself. It is necessary to understand which browsers are available, what are their capabilities, and how they compare to desktop browsers. On mobile, more than 70%[3] of current mobile traffic comes from Webkit browsers (essentially Safari Mobile and the Android browser), and because Webkit represents a larger percentage of mobile browsers we can expect similar web rendering on simple pages (Firtman 2010, 44), which is good in terms of web development; however, there are differences between Webkit implementations, so testing only on one device and browser does not guarantee that a page works perfectly on every Webkit instance. Mobile browsers are in general more limited in terms of their features than their desktop counterparts, but more advance than older desktop browsers[4].

Furthermore, mobile devices are also more limited in terms of performance, which is easier to note with complex animations on slower devices. Likewise, the network performance should not be disregarded. Mobile devices can be used in outdoor environments, where the only Internet access available is through mobile networks that are slower, and generally have expensive data plans. That is reflected on the need to reduce and optimize the assets that are served to mobile devices.

The aforementioned topics may impose considerable limitations on the design of websites for the mobile landscape, but are those limitations that make the mobile web a much more changeling and interesting subject within the field of web design. There is a more open and receptive space to novelty interfaces, which leads us to the opportunity to find and propose new patterns.

3. PATTERN MODEL

To help us finding a template for the patterns in this work, and to better understand the writing of patterns, we started our work by developing a comparative analysis among sixteen libraries[5]. This previous work allowed us to better justify the template that is proposed for the organization of the patterns in this work. We end up with a format that resembles the one used in Alexander's patterns (1977), with a few differences. In short, each

[3] According to the data from StatCounter from the first trimester of 2012, when we take as reference Europe or the USA and combine the results from the major Webkit browsers.
[4] We are referring to browsers that were released years ago but are sill used today, e.g. Internet Explorer 6.
[5] A Pattern Language (Alexander, Ishikawa and Silverstein 1977), Design Patterns: Elements of Reusable Object-Oriented Software (Gamma et al. 1995), Common Ground (Tidwell 1999), A Pattern Approach to Interaction Design (Borchers 2001), The Design of Sites: Patterns for Creating Winning Websites (van Duyne, Landay and Hong 2006), Designing Interfaces (Tidwell 2011), Yahoo! Design Pattern Library (Yahoo! 2006), Patterns for Computer-Mediated Interaction, Info Design Patterns (Schümmer and Lukosch 2007), Patternry (Factory 2009), Designing Web Interfaces (Scott and Neil 2009), Designing Social Interfaces (Crumlish and Malone 2009), UI Patterns (Toxboe), Banco de Padrões de Design (IST 2010), Designing Mobile Interfaces (Hoober and Berkman 2011), Mobile Design Pattern Gallery (Neil 2012)

pattern includes in the following order: name, illustration, problem, solution, rationale, examples, related patterns (complementary patterns in this library and similar pattern in other libraries).

Patterns are organized with an implicit hierarchy that can be inferred by the order in which they appear on this paper. They are sorted in terms of scale, from the ones that address the macro structure of a design, to ones related to interaction details.

We formatted patterns with a highlighted block, composed by the name, problem, solution and illustration that work as a summary of the pattern. We devise the pattern model in a way that the reading of this first block should provide an overview of the pattern. The problem and solution statements were written to be read in sequence; it was posed as: *if* problem *then* solution. However, reading this first block it is enough for a successful implementation of the pattern.

A major difference between the format proposed in this work, and others that are used in libraries that dealt with interaction design is the effort given to the Illustration section. These illustrations use a simplified and coherent graphic language — analogous to the one used by Hoober and Berkman (2011) —, with the purpose of removing irrelevant artifacts that a screenshot of a real website may contain, and to make them easier to compare. Besides the main image that illustrates each pattern we developed a complementary interactive illustration for each one, in a hypothetical website — patterns.jribeiro.org — that is used to demonstrate the patterns in the library. We expect that the static illustration that follows each pattern in this paper is enough for its understanding; however, we hope that the interactive illustrations became a valuable asset that provides to the reader with a better and precise representation of how a pattern really works.[6]

4. WEB DESIGN PATTERNS FOR MOBILE DEVICES

The idea, and sometimes the name, of the patterns included in this paper came from two main sources: from other pattern libraries with comparable patterns, and from the research on designing websites for mobile; complemented with the work on real design projects.

In beginning of this project, we conducted a comparative analysis between libraries with patterns for interaction design, particularly those that already addressed the design of mobile interfaces. In this analysis we collected patterns that could be adopted and adapted to the design of mobile websites, and rewrote them to respond to the particularities that designing for the web on smaller screens impose. For example, the pattern VERTICAL LIST has a comparable pattern in almost all libraries that were analyzed. It appears as *List Menu* in the *Mobile Design Pattern Gallery* (Neil 2012) or as *Vertical List* in *Designing Mobile Interfaces* (Hoober and Berkman 2011). Others came to life from reading and researching on the design of interfaces for mobile devices. For instance, the pattern ICEBERG TIP was inspired on the idea presented in *Designing Gestural Interfaces* (Saffer 2008). Other patterns, such as TABS and DROPDOWN, which are quite common patterns on the web, were adapted in order to respond to the limitations that these devices impose. actions

[6] These interactive illustrations allows us to view how the pattern adapts to different sizes. We can toggle between portrait, landscape and desktop views.

4.1 Linearized Layout

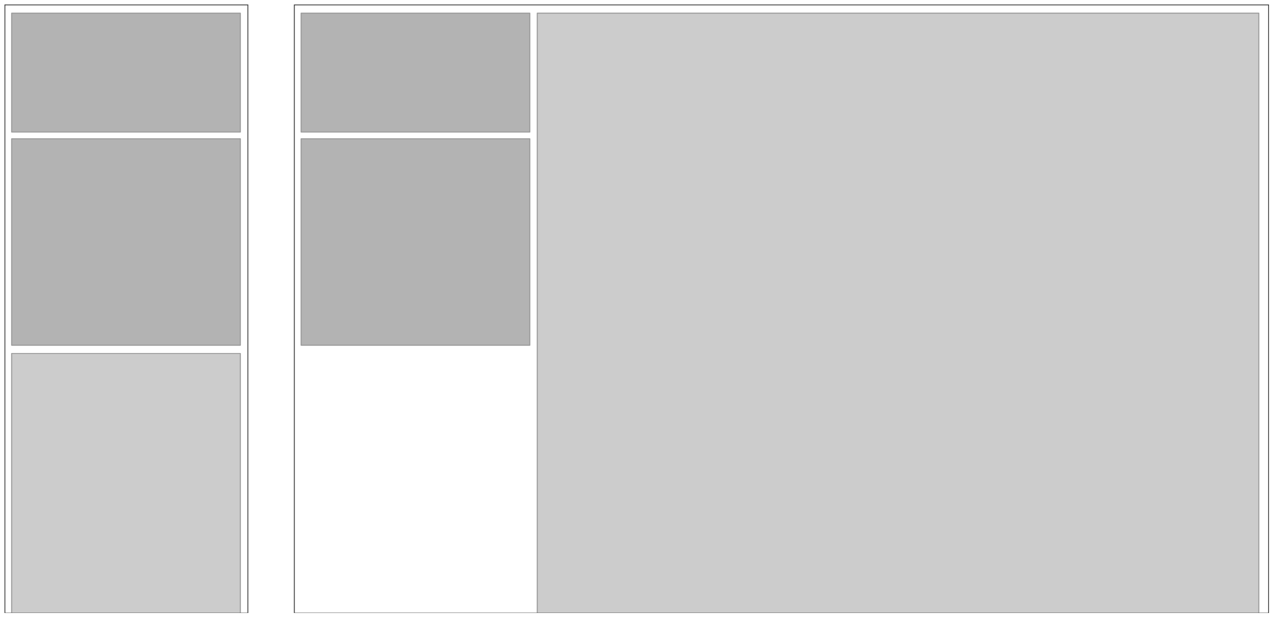


Figure 1. LINEARIZED LAYOUT.

Problem

On smaller devices with narrow screens, more complex designs with multiple columns do not fit perfectly. In most cases you end up with a scaled-down page that is unreadable.

Solution

Linearize the content, by stacking all blocks of information on top of each other, spanning the width of the screen.

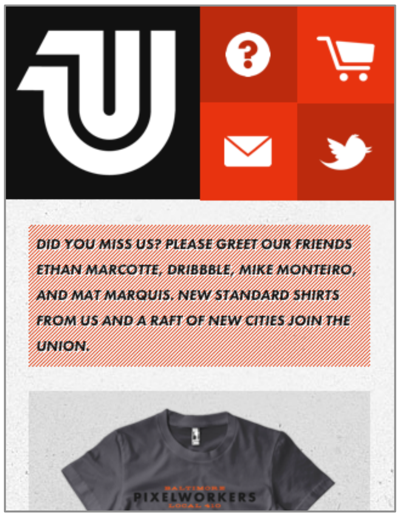
***

Layouts with multiple columns are a common pattern on web design. However, they do not work properly on devices with narrow viewports. When this type of websites are accessed on these devices, browsers will zoom out the page until it fits on the width of the screen; or they will add horizontal scrolling. Both alternatives are far from satisfying. Column layouts should be optimized for mobile viewing. Therefore, you should design your mobile site by assembling the page vertically and by stacking each page section on top of each other, which is already the default behavior of a page when no style is defined.

Because we cannot foresee the size and resolution of the devices that will access our website, blocks with fixed widths are not a good practice. The LINEARIZED LAYOUT should have a fluid layout that adapts to the width of the browser whatever is its effective size.

This pattern is also a requirement for many of the other patterns proposed, particularly VERTICAL LIST, i.e., if you want to use it you will invariably need to linearize the content.

Examples


DID YOU MISS US? PLEASE GREET OUR FRIENDS ETHAN MARCOTTE, DRIBBBLE, MIKE MONTEIRO, AND MAT MARQUIS. NEW STANDARD SHIRTS FROM US AND A RAFT OF NEW CITIES JOIN THE UNION.

http://www.unitedpixelworkers.com

http://colly.com
http://ucsd.edu
http://foodsense.is
http://cironline.org

Related Patterns

– GRID LAYOUT
– VERTICAL LIST

– *Vertical Stack* in *Designing Interfaces* (Tidwell 2011)

4.2 Grid Layout

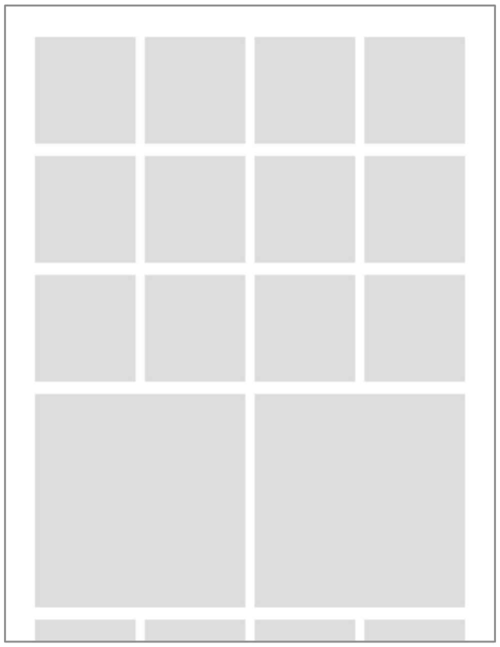


Figure 2. GRID LAYOUT.

Problem

Although a LINEARIZED LAYOUT works in most cases, sometimes you have content that does not need to, or should not, fill the entire width of screen.

Solution

Arrange elements in a matrix of one or more rows.

***

While not as recurrent as the LINEARIZED LAYOUT, a GRID LAYOUT can be quite effective with some types of content. It is typically used with image content such as photos, illustrations or icons. The most common case is to present a gallery of images, but you can still use it with text, as long you keep it to a few words. You can still use it with text, as long you keep it to a few words. Long paragraphs of text can become almost unreadable even if your grid only has two columns.

A GRID LAYOUT can have as many divisions as needed; however, if the grid blocks work as buttons they should be large enough and adequately spaced so they are easily triggered — use a TOUCH FRIENDLY BUTTON or an ICEBERG TIP on those cases. Although in most examples blocks have the same height and width, they can still assume irregular forms. That is, blocks can span multiple columns or rows — Figure 3.

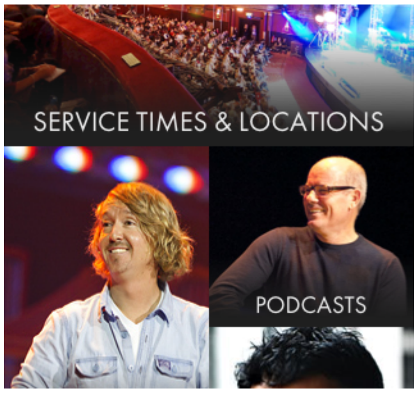


Figure 3. Hillsong website.

Because with this pattern you are displaying multiple items side by side, in some cases it can be a more efficient use of the vertical space than a LINEARIZED LAYOUT. For example, if you have a VERTICAL LIST where each item is composed by only one word, you can save space by displaying several items on the same line.

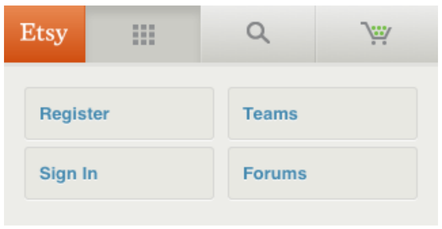


Figure 4. The TOGGLE MENU on Etsy website reveals a menu that list items on a grid.

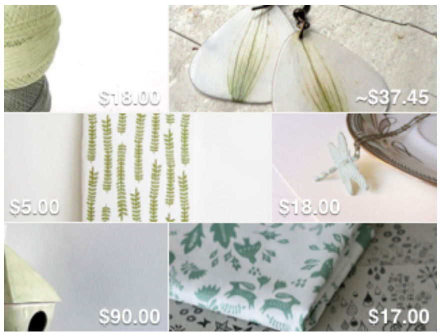
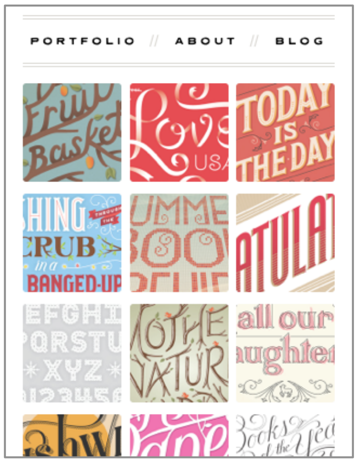


Figure 5. The Etsy website uses a SLIDESHOW that presents on the same slide multiple images on grid, each one working as a link.

The GRID LAYOUT can be used as a more structural pattern, or combined with other patterns to extend their capabilities. For example it can be used with a TOGGLE MENU to increase the number of items visible on the screen — Figure 4; or with a SLIDESHOW to increase the number of items by slide — Figure 5.

Examples



http://jessicahische.is,

http://m.microsoft.com
http://2012.dconstruct.org

Related Patterns

– LINEARIZED LAYOUT
– TOGGLE MENU
– SLIDESHOW

– *Grid* in *Designing Mobile Interfaces* (Hoober and Berkman 2011)

4.3 Vertical List

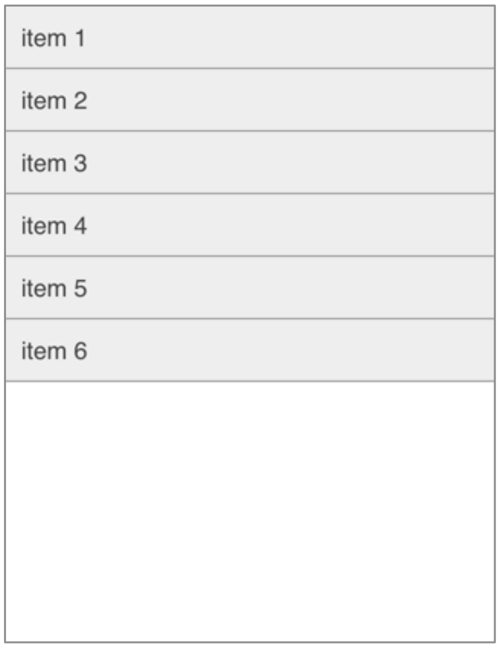


Figure 6. VERTICAL LIST

Problem

A LINEARIZED LAYOUT is a quite common layout that gives much emphasis to the vertical orientation. If you have information that is organized as a list, you need to have a method to efficiently display that content.

Solution

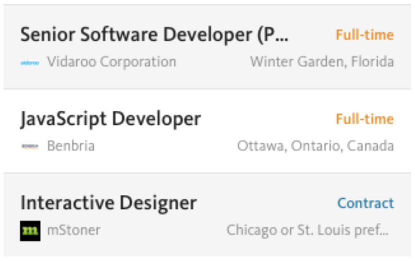
Display these chunks of information stacked vertically, spanning all the width of the screen and graphically dividing each item.

***

Along with the LINEARIZED LAYOUT this is perhaps the most common pattern that you will encounter when designing for mobile. Given that the width of the devices is not generous, you will invariably need to rely on the vertical space to accommodate most of the content. For most of these situations, this pattern or one of its variations will be used, such as: INFINITE LIST, THUMBNAIL LIST or EXPANDING LIST. Therefore, most of the recommendations given for this pattern also apply to the related patterns.

Generally, there are two different alternatives to the implementation of this pattern: one that only displays information; another where each list item works as a link — which sometimes can be a LINEARIZED MENU.

When the list items work as links you should optimize them for touch, you can use a TOUCH FRIENDLY TARGET for this. Each item ideally should only contain a link. Even if the item contains text with different hierarchies — Figure 7 —, you can wrap all those elements on an anchor tag rather than using only the title as a link. The result is similar to what can be attained with an ICEBERG TIP.


Senior Software Developer (P... Full-time
Vidaroo Corporation Winter Garden, Florida

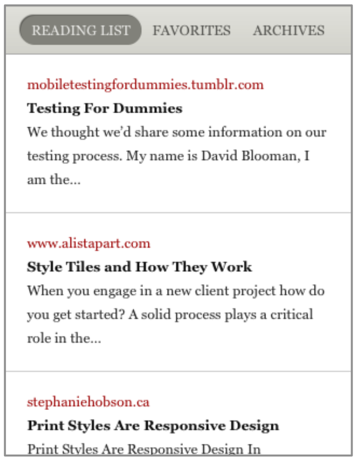
JavaScript Developer Full-time
Benbria Ottawa, Ontario, Canada

Interactive Designer Contract
mStoner Chicago or St. Louis pref...

Figure 7. On the *Authentic Jobs* website, while each list item contains a diverse range of information with different hierarchies, the entire rectangle works as a link.

For this pattern to work effectively, each item should be clearly separated. You can use any design that visibly distinguishes items, such as: a horizontal line spanning the width of the screen; alternative color rows; typographic hierarchy; or just white space.

Examples



mobiletestingfordummies.tumblr.com
Testing For Dummies
We thought we'd share some information on our testing process. My name is David Blooman, I am the...

www.alistapart.com
Style Tiles and How They Work
When you engage in a new client project how do you get started? A solid process plays a critical role in the...

stephaniehobson.ca
Print Styles Are Responsive Design
Print Styles Are Responsive Design In

http://readability.com/mobile

http://cognition.happycog.com/topics
http://bostonglobe.com

Related Patterns

– INFINITE LIST
– THUMBNAIL LIST
– EXPANDING LIST
– LINEARIZED MENU

– *Vertical* List in *Designing Mobile Interfaces* (Hoober and Berkman 2011)
– *List Menu* in *Mobile Design Pattern Gallery* (Neil 2012)

4.4 Infinite List

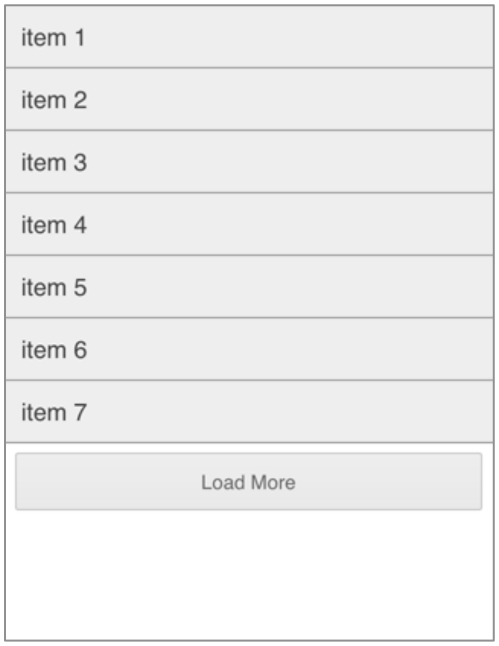


Figure 8. LINEARIZED MENU

Problem

A very extensive VERTICAL LIST can become quite heavy on mobile devices. It would not be advisable to load and display all information in the list at once.

Solution

Design a normal VERTICAL LIST or any of its variations but only fetch the beginning of the list, loading the rest as the user scrolls through the page.

***

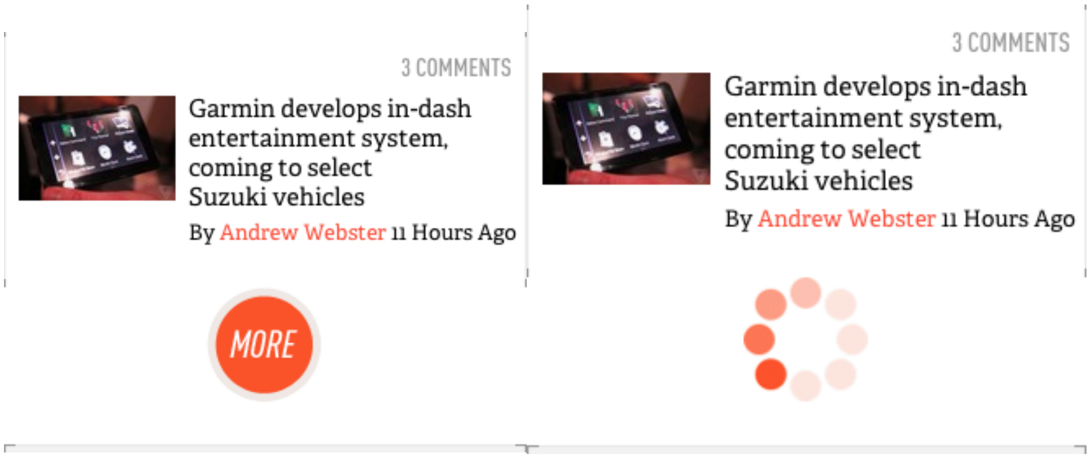
This pattern is based on a homonym pattern from *Designing Mobile interfaces* (Hoober and Berkman 2011). It is very similar to the VERTICAL LIST with the main difference that only a portion of the list is initially loaded. It is most adapted for displaying a list of data with an uncountable number of items. For example, when loading all the news of a site in the same page we can be dealing with hundreds of items. It would not be a good idea to load all that information at once. That would be extremely heavy in terms of download size and load time, and probably, users would not need all that information from the beginning. Thus, we can start by loading only a few items, and only load subsequent items when the user provides some signal that he wants to keep browsing through the page. The number of items to load on each action varies, but it will depend on the length and download size of the content.

This pattern takes advantage of methods for asynchronously loading additional content without the need to refresh the page: new items are just appended to the interface following the previous ones. Because the browser does not need to reload the page, we can get a perceivably faster response. It can be used as an alternative to a common pagination, which normally implies more touches and a refresh on each load. Since there is not any refresh, users never lose the context of where they are at the list.

There are two main alternatives that can be used for its implementation: explicit and implicit loading. On the first, the content is loaded with a direct action of the user; on the second, the list is loaded automatically as the user reaches its end.

When designing a list in which the user has to explicitly load new content, place at the bottom of the list a button that indicates that more data will be loaded on the same page. You can use a label with "more", "load more", or another appropriate variation. You can also use that button to indicate how many more items will be loaded.

While the browser is loading the next chunk of the list, provide some feedback of the action that is occurring. In Figure 9, the page displays a 'More' button that when pressed changes to a loading animation.

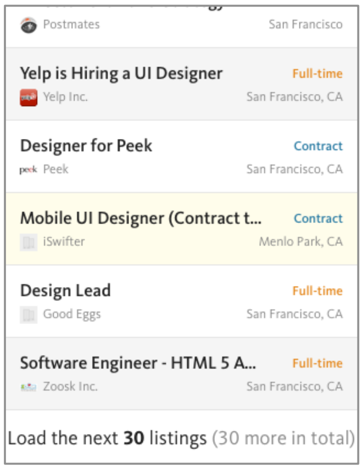

3 COMMENTS
Garmin develops in-dash entertainment system, coming to select Suzuki vehicles
By Andrew Webster 11 Hours Ago
MORE
3 COMMENTS
Garmin develops in-dash entertainment system, coming to select Suzuki vehicles
By Andrew Webster 11 Hours Ago
Figure 9. Loading animation at *The Verge* (www.theverge.com).

In the implicit loading mode there is not any direct action of the user to load additional content. Instead, the browser detects when a user reaches the end of the page and automatically loads another portion of the list. This is a type of implementation that is normally called *lazy loading*. To give the impression that the list is really bottomless, you can start preloading the adjacent content before the user gets to end of the list.

Although this pattern is normally used with content that is virtually endless, it is still possible for users to reach its end. Therefore, provide some indication when there are no more items to fetch.

Because this pattern is a variation of the VERTICAL LIST, it can be combined with any of its other variants, such as: THUMBNAIL LIST, EXPANDING LIST.

Examples



Postmates	San Francisco
Yelp is Hiring a UI Designer	Full-time
Yelp Inc.	San Francisco, CA
Designer for Peek	Contract
Peek	San Francisco, CA
Mobile UI Designer (Contract t...	Contract
iSwifter	Menlo Park, CA
Design Lead	Full-time
Good Eggs	San Francisco, CA
Software Engineer - HTML 5 A...	Full-time
Zoosk Inc.	San Francisco, CA

Load the next **30** listings (30 more in total)

http://www.authenticjobs.com

http://theverge.com
http://facebook.com
http://gmail.com

Related Patterns

– VERTICAL LIST
– THUMBNAIL LIST
– EXPANDING LIST

– *Infinite List*, in *Designing Interfaces* (Tidwell 2011)
– *Infinite List*, in *Designing Mobile Interfaces* (Hoober and Berkman 2011)

4.5 Thumbnail List

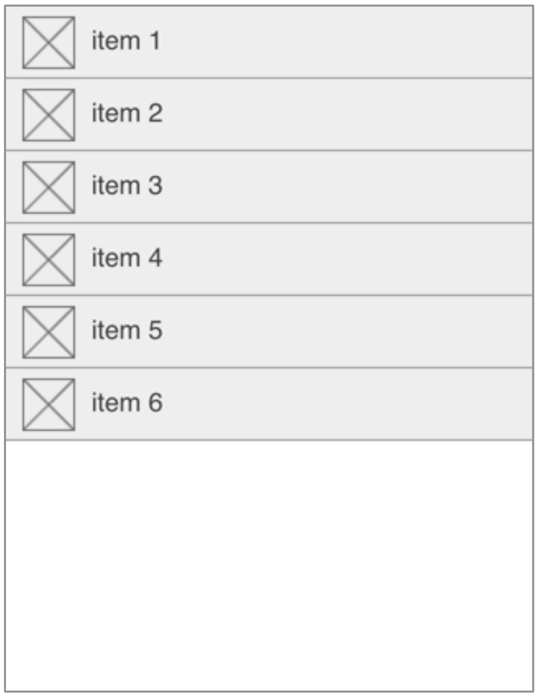


Figure 10. THUMBNAIL LIST.

Problem

It can be difficult to find a particular item on a VERTICAL LIST that is composed only by text because all items may look identical. You need to make each list item more distinct so that the list is easier to scan.

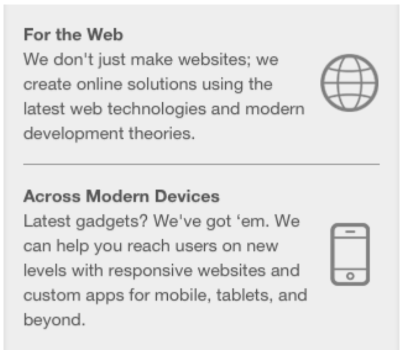
Solution

Design a VERTICAL LIST where in addition to the textual information, you also display a small illustration next to each list item.

***

An extensive list of items composed only with text can be visually monotonous and harder to scan. You can minimize this problem by complementing each list item with a thumbnail-size image that is illustrative of the content. Because each image can have distinct shapes and colors, they are easier to scan and interpret. The image should somehow be related to the content of the item, but you can use either a photo or an icon. This pattern is based on the patterns *Thumbnail List* (Hoober and Berkman 2011), from where it got its name, or *Thumbnail-and-Text List* (Tidwell 2011).

Thumbnails are usually aligned to the left. However, if images are optional, you can align them to the right so you can create a better defined axis — Figure 11. You can use a placeholder image for instances where images are not available, but keep in mind that if most images are placeholders the benefits of the THUMBNAIL LIST are lost.

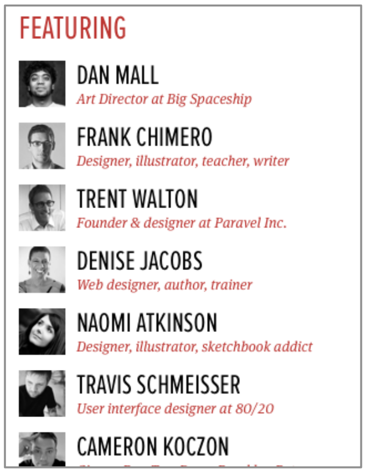

For the Web
We don't just make websites; we create online solutions using the latest web technologies and modern development theories.

Across Modern Devices
Latest gadgets? We've got 'em. We can help you reach users on new levels with responsive websites and custom apps for mobile, tablets, and beyond.

Figure 11 Right aligned THUMBNAIL LIST at *Meltmedia* website.

This pattern can be used with the other variations of the VERTICAL LIST, such as the INFINITE LIST and the EXPANDING LIST.

Examples


FEATURING

DAN MALL
Art Director at Big Spaceship

FRANK CHIMERO
Designer, illustrator, teacher, writer

TRENT WALTON
Founder & designer at Paravel Inc.

DENISE JACOBS
Web designer, author, trainer

NAOMI ATKINSON
Designer, illustrator, sketchbook addict

TRAVIS SCHMEISSER
User interface designer at 80/20

CAMERON KOCZON

http://2012.newadventuresconf.com

http://mobile.theverge.com
http://mobile.engadget.com

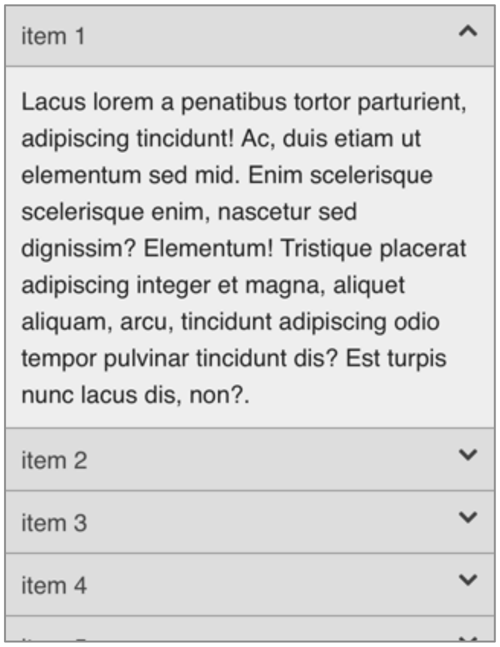
http://poetfreak.com/explore
http://meltmedia.com

Related Patterns

– VERTICAL LIST
– INFINITE LIST
– EXPANDING LIST
– LINEARIZED MENU

– *Thumbnail List* in *Designing Mobile Interfaces* (Hoober and Berkman 2011)
– *Thumbnail-and-Text List* in *Designing Interfaces* (Tidwell 2011)

4.6 Expanding List


item 1

Lacus lorem a penatibus tortor parturient, adipiscing tincidunt! Ac, duis etiam ut elementum sed mid. Enim scelerisque scelerisque enim, nascetur sed dignissim? Elementum! Tristique placerat adipiscing integer et magna, aliquet aliquam, arcu, tincidunt adipiscing odio tempor pulvinar tincidunt dis? Est turpis nunc lacus dis, non?.

item 2

item 3

item 4

Figure 12. EXPANDING LIST

Problem

You need to display a series of related information that has a clearly defined hierarchy. However, vertically displaying all that information would lead to a very long page.

Solution

Design a VERTICAL LIST or one of its variations, but display only part of the content — usually the heading — as a toggle to show additional content.

***

This pattern got its name from a similar pattern in *Mobile Design Pattern Gallery* (Neil 2012) and is a variation of the VERTICAL LIST, in which the visible item does not present static information or works as a link to another page, but is rather used to trigger the visibility of additional content in the same page. Tapping on the visible part of the item makes it expand, revealing the hidden content. It is most suitable for when you need to present content with a clearly defined hierarchy; for example, when you are designing a LINEARIZED MENU with subitems.

Although it is possible to present more than two levels of information with this pattern, it can be confusing to do so.

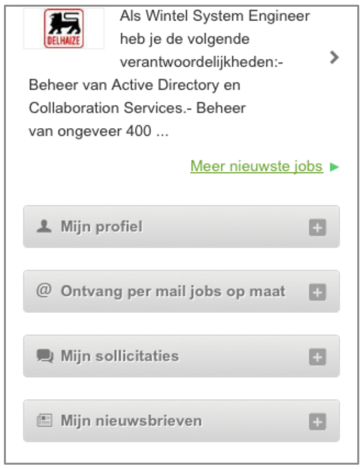
You should provide some clues to indicate that additional content is available. For example, a downward arrow that changes to an upward arrow when the item is expanded; or a plus sign that changes to a minus sign. You can give emphasis to the fact that the item has expanded by implementing a small animation showing the content appearing.

Besides clearly distinguish between list items — as it is described on the VERTICAL LIST —, you should also differentiate between the heading of the item and the respective content. More important, you should design them so that the revealed content is grouped to the upper heading rather than the order way around.

Interactions Details

In terms of the behavior of the list there are two alternatives for the implementation of this pattern: one that works as a toggle; another that works as an accordion. In the toggle type each item works independently, that is, regardless of the state of all other items in the list, when you tap on one it expands, when you tap it again it collapses. In the accordion type, elements of the list are connected, when the user taps on the header of the item the content of that item is expanded and all others are collapsed. These two different behaviors are sometimes described as different patterns, for example, in *Designing Interfaces* (2011), Tidwell presents the patterns, *Accordion*, and *Collapsible Panels*.

Examples


Als Wintel System Engineer heb je de volgende verantwoordelijkheden:- Beheer van Active Directory en Collaboration Services.- Beheer van ongeveer 400 ...
Meer nieuwste jobs
Mijn profiel
Ontvang per mail jobs op maat
Mijn sollicitaties
Mijn nieuwsbrieven
http://www.jobat.be/nl

http://www.m.microsoft.com
http://www.unicefusa.org/mobile

Related Patterns

– VERTICAL LIST
– INFINITE LIST
– THUMBNAIL LIST
– LINEARIZED MENU

– *Expanding List*, in M*obile Design Pattern Gallery* (Neil 2012)
– *Windowshade* in *Designing Mobile Interfaces* (Hoober and Berkman 2011)
– *Accordion*, in *Designing Interfaces* (Tidwell 2011)
– *Collapsible Panels*, in *Designing Interfaces* (Tidwell 2011)

4.7 Linearized Menu

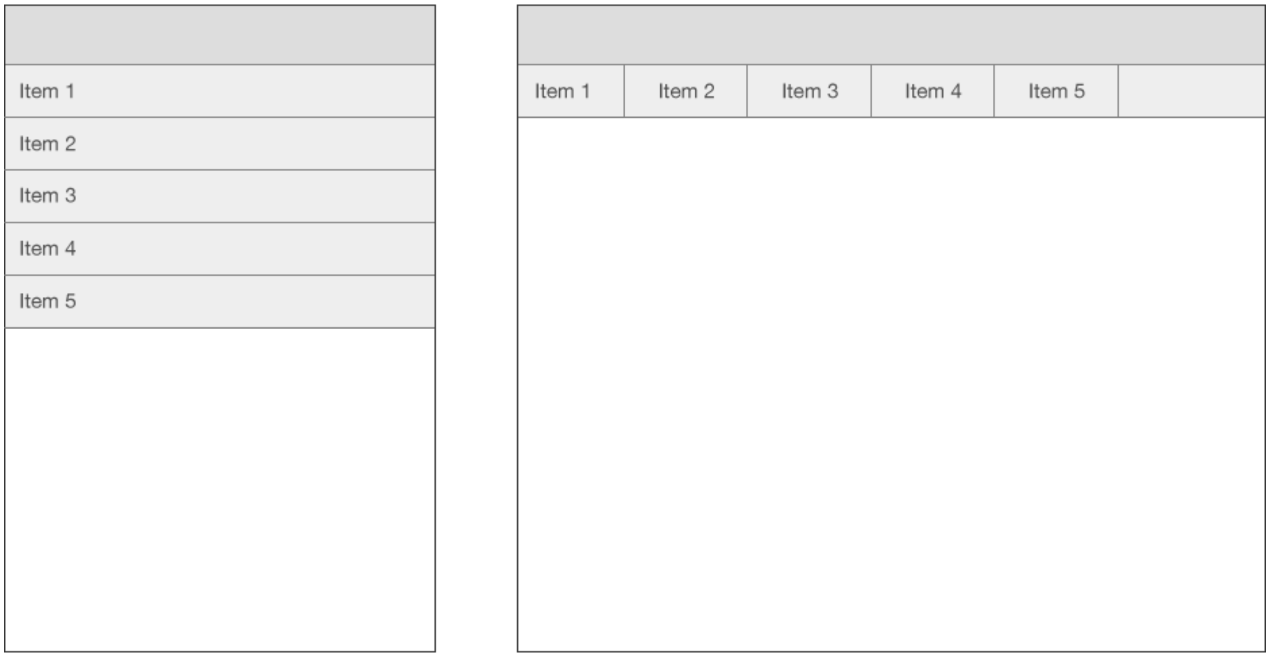


Figure 13. LINEARIZED MENU, mobile and desktop versions.

Problem

Although inline menus are quite common on desktop websites, they are difficult to achieve on mobile. If your menu has several items, it will not fit properly on the mobile version of the website.

Solution

List all menu items vertically spanning the width of the device.

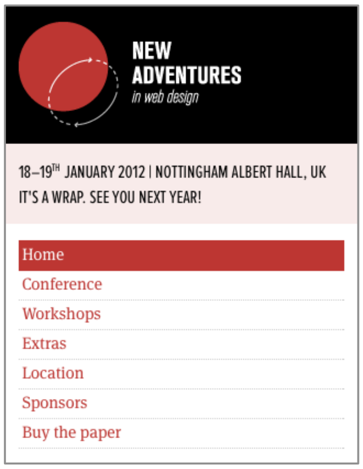
***

Because of the narrow width of mobile devices, in most cases you do not have enough space to display items inline. The solution involves disposing list items vertically covering all the width of the screen. This type of menu is quite easy to implement because it is the default behavior of a list when not styled.

Be aware that if the menu is extensive and is positioned on top of the page it will probably fill most of the page. You can easily cope with this problem with a JUMP MENU.

You should optimize list items for touch by making the list span across the width of the screen, and with height enough so they are easily triggered and nicely spaced so the wrong target is not tapped by mistake — TOUCH FRIENDLY BUTTON or ICEBERG TIP.

Examples



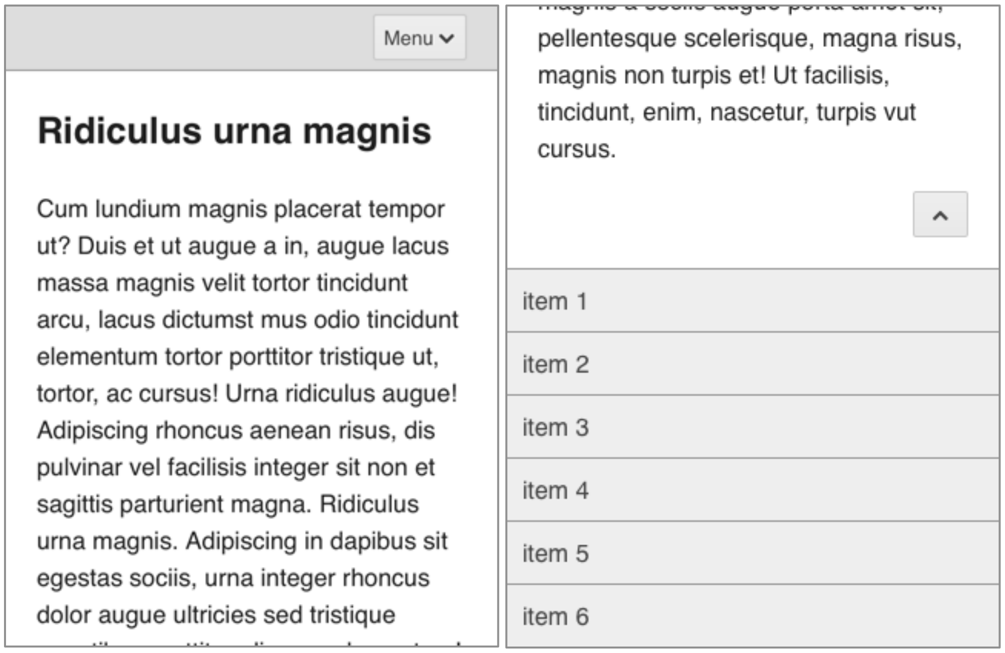
http://2012.newadventuresconf.com

http://clearairchallenge.com
http://cacaotour.com

Related Patterns

– VERTICAL LIST
– JUMP MENU
– TOGGLE MENU
– SIDE MENU

4.8 Jump Menu



Cum lundium magnis placerat tempor ut? Duis et ut augue a in, augue lacus massa magnis velit tortor tincidunt arcu, lacus dictumst mus odio tincidunt elementum tortor porttitor tristique ut, tortor, ac cursus! Urna ridiculus augue! Adipiscing rhoncus aenean risus, dis pulvinar vel facilisis integer sit non et sagittis parturient magna. Ridiculus urna magnis. Adipiscing in dapibus sit egestas sociis, urna integer rhoncus dolor augue ultricies sed tristique

pellentesque scelerisque, magna risus, magnis non turpis et! Ut facilisis, tincidunt, enim, nascetur, turpis vut cursus.

Figure 14. JUMP MENU.

Problem

When placed on top of the page, an extensive LINEARIZED MENU will fill all the available space of the screen. However, generally, you do not want the menu to take precedence over the content.

Solution

Place the menu at the bottom of the page but display a button on top of the page that links to the menu.

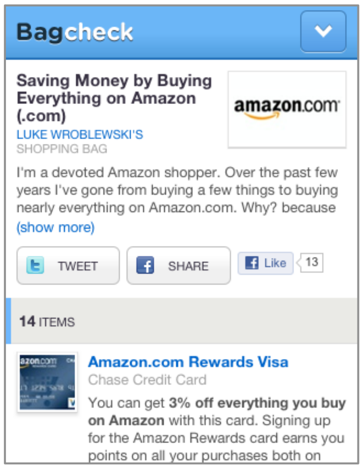
***

A navigation menu with an extensive list of items can fill the entire screen when the page loads, relegating the content to second place. However, it is usually a good practice to emphasize content over navigation (Wroblewski 2011). This pattern tries to overcome this problem by focusing on the content while still providing quick access to the navigation. For that, you design a LINEARIZED MENU that is positioned at the bottom of the page while leaving a button on top that takes the user to the menu. Besides the button on top, it is helpful to provide a link next to the menu to take users back to the top, so they do not need to scroll the entire page if they need to go back.

Since the JUMP MENU only uses a normal HTML anchor and there is no JavaScritp required, it is extremely simple to implement and probably will work with almost all browsers and devices.

The jump to the footer can be disorienting because the screen abruptly changes from one state to another without much feedback of what happened. You can decrease the problem caused by the sudden jump by using an animation that scrolls through the entire page until the menu. However, depending on the length of the page and how the animation is done, you can be unnecessarily delaying the access of the user to the menu. Moreover, this type of animations can be sluggish on slower devices.

Examples


Bagcheck
Saving Money by Buying Everything on Amazon (.com)
LUKE WROBLEWSKI'S
SHOPPING BAG
amazon.com
I'm a devoted Amazon shopper. Over the past few years I've gone from buying a few things to buying nearly everything on Amazon.com. Why? because
(show more)
TWEET
SHARE
Like
13
14 ITEMS
Amazon.com Rewards Visa
Chase Credit Card
You can get 3% off everything you buy on Amazon with this card. Signing up for the Amazon Rewards card earns you points on all your purchases both on

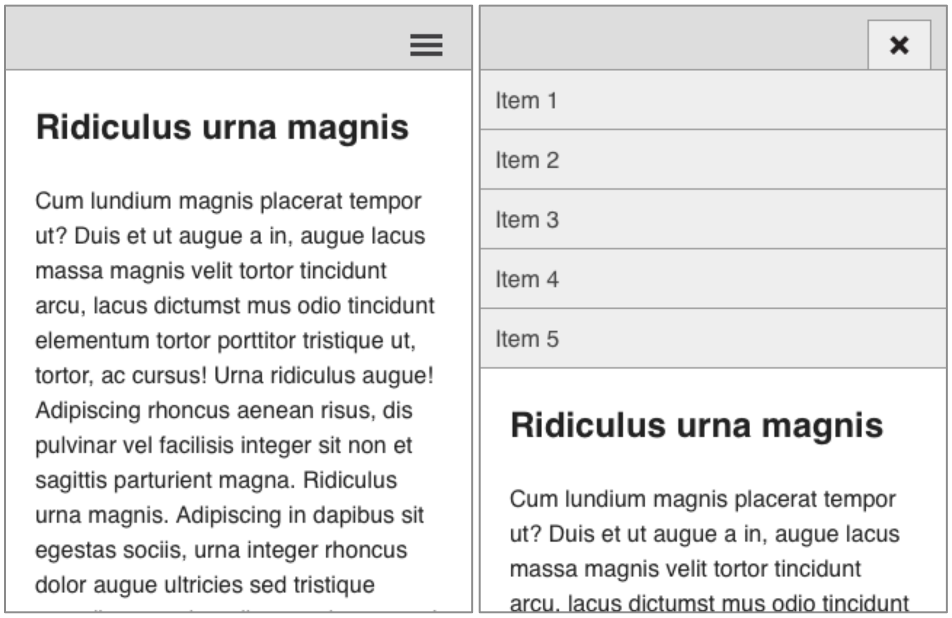
https://bagcheck.com

http://unicef.se
http://kiskolabs.com
http://bearded.com
http://contentsmagazine.com
http://builtwithmomentum.com

Related Patterns

– LINEARIZED MENU
– TOGGLE MENU

4.9 Toggle Menu



Ridiculus urna magnis

Cum lundium magnis placerat tempor ut? Duis et ut augue a in, augue lacus massa magnis velit tortor tincidunt arcu, lacus dictumst mus odio tincidunt elementum tortor porttitor tristique ut, tortor, ac cursus! Urna ridiculus augue! Adipiscing rhoncus aenean risus, dis pulvinar vel facilisis integer sit non et sagittis parturient magna. Ridiculus urna magnis. Adipiscing in dapibus sit egestas sociis, urna integer rhoncus dolor augue ultricies sed tristique

Ridiculus urna magnis

Cum lundium magnis placerat tempor ut? Duis et ut augue a in, augue lacus massa magnis velit tortor tincidunt arcu, lacus dictumst mus odio tincidunt

Figure 15. TOGGLE MENU.

Problem

You have an extensive LINEARIZED MENU that takes the entire screen, and a JUMP MENU is not a proper alternative because it displaces the menu to the bottom of the page. You want to display the menu on the top of the page but do not want it to fill the entire screen.

Solution

Design the menu content as a LINEARIZED MENU but conceal it, then provide a button to toggle the visibility of the menu.

***

In the TOGGLE MENU we have a LINEARIZED MENU that is presented collapsed when the page loads until there is a direct action of the user to expand it. This allows us to display only a small button on top of the page to toggle the visibility of the menu. With this approach we can present the content first while still providing quick access to the navigation.

This pattern is somehow similar to the SELECT MENU in that, a link to the navigation is presented in the top of the site, and the navigation is only disclosed when there is a direct instruction of the user. The main difference between both is that the SELECT MENU uses the native select menu component of the device while this pattern uses a custom interface. This is a much more clean and elegant approach to the same problem and should rather be used instead of the SELECT MENU whenever it is possible.

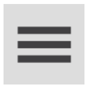


Figure 16. A possible toggle icon.

For the element which triggers the menu you can use any symbol that provides the correct affordance that additional information will be disclosed. Nonetheless, many websites use an icon with three horizontal bars, which represent the list items of a menu — Figure 16. When the menu is active you can change the icon to show that the menu state has changed. For example, the Starbucks website changes the icon to an “x” when the menu is expanded. Andy Clarke (Clarke 2012) incites the need to reach a consensus on a standard icon for showing navigation, settling his support for the three lines because they are widely used and therefore, easily recognized, unless your navigation is arranged on a grid, which, in that case you should use a grid icon. In short, the symbol used should map the layout of the menu.

If you have enough horizontal space you can improve the usability of this pattern by appending the title of the current page to the toggle icon — Figure 17 This allows us to give feedback of the user’s position in the site hierarchy and provide a larger target.

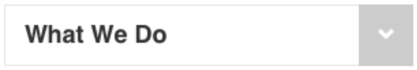
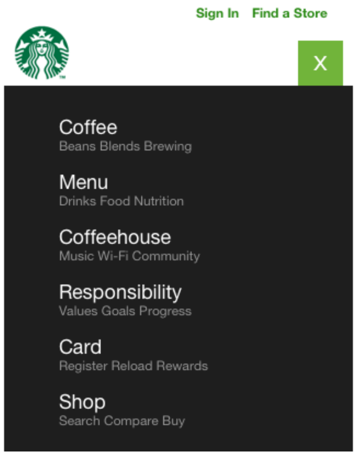


Figure 17. *Filament Group* approach to this pattern uses the toggle button to display the title of the current page.

Examples



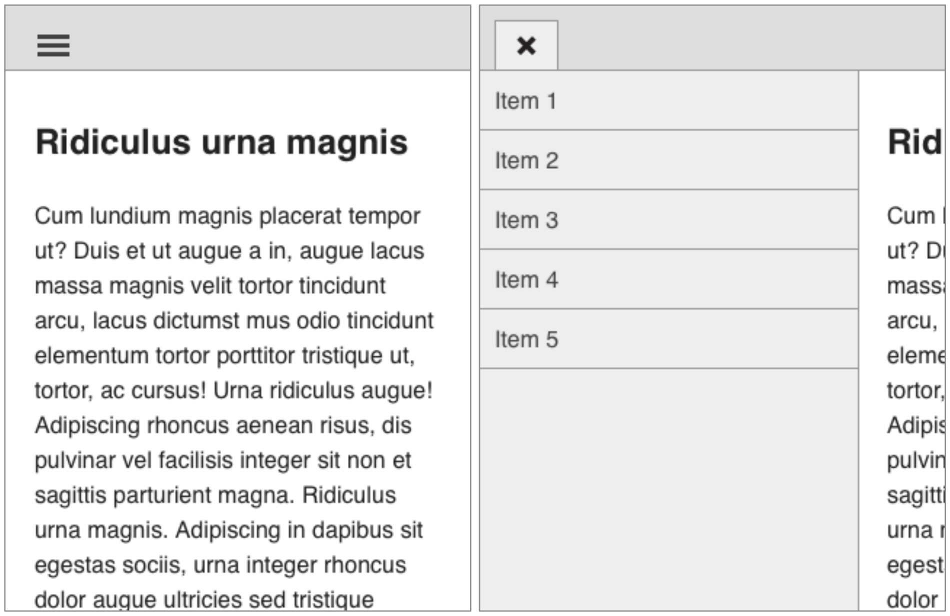
http://starbucks.com

http://twitter.github.com/bootstrap
http://m.bbc.co.uk/news
http://filamentgroup.com/examples/rwd-nav-patterns
http://m.nfl.com
http://brickartist.com

Related Patterns

– EXPANDING LIST
– SELECT MENU
– DROPDOWN MENU

4.10 Side Menu



Ridiculus urna magnis

Cum lundium magnis placerat tempor ut? Duis et ut augue a in, augue lacus massa magnis velit tortor tincidunt arcu, lacus dictumst mus odio tincidunt elementum tortor porttitor tristique ut, tortor, ac cursus! Urna ridiculus augue! Adipiscing rhoncus aenean risus, dis pulvinar vel facilisis integer sit non et sagittis parturient magna. Ridiculus urna magnis. Adipiscing in dapibus sit egestas sociis, urna integer rhoncus dolor augue ultricies sed tristique

Figure 18. SIDE MENU

Problem

Although vertical menus side by side with the main content are fairly common, they are almost impossible to achieve on a mobile device. It would not be efficient to reserve an entire column on a small screen just for the menu. Nonetheless, you may still want to get a similar look.

Solution

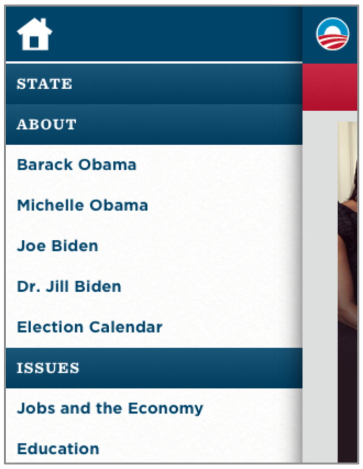
Design the menu bonded together to one side of the page, but positioned outside the page. Then, provide a button that will show the menu by sliding it in, sliding out the content.

***

Vertical navigation placed side by side with the page's main content is a quite common pattern on websites. On mobile, that type of navigation is not practical because horizontal space is limited, but this pattern allows us to achieve a comparable layout on both versions. Like the TOGGLE MENU, this pattern allows us to focus on the content while providing quick access to the navigation.

The idea of this pattern was first formulated as a pattern by Frost (2012b) as *The Left Nav Flyout*, and consists of a button on top of the page that allows users to toggle the visibility of a hidden menu. When that button is tapped it reveals the menu on one of the side of the screen by pushing the main content out of the screen. You should keep a small portion of the page to give some affordance of how the menu works. Additionally, you can display an animation of the menu moving to show users what is happening. Like in the JUMP MENU, an abrupt change of the context can disorient the user.

Examples



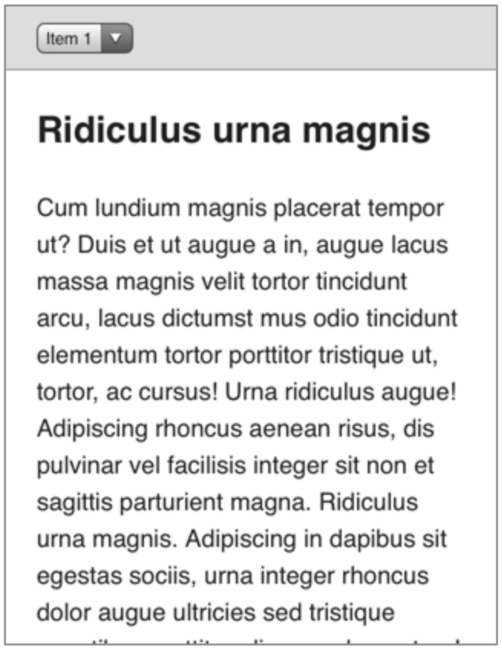
http://www.barackobama.com

http://facebook.com
http://www.breakingnews.com/submitted
http://kettlenyc.com
http://fortysevenmedia.com

Related Patterns

– TOGGLE MENU

4.11 Select Menu



Ridiculus urna magnis

Cum lundium magnis placerat tempor ut? Duis et ut augue a in, augue lacus massa magnis velit tortor tincidunt arcu, lacus dictumst mus odio tincidunt elementum tortor porttitor tristique ut, tortor, ac cursus! Urna ridiculus augue! Adipiscing rhoncus aenean risus, dis pulvinar vel facilisis integer sit non et sagittis parturient magna. Ridiculus urna magnis. Adipiscing in dapibus sit egestas sociis, urna integer rhoncus dolor augue ultricies sed tristique

Figure 19. SELECT MENU.

Problem

On small screens, an extensive menu can fill the entire page. However, if you have a menu that has to work simultaneously on wide screens and on mobile devices, you want that menu to be as efficient as possible regarding the use of vertical space.

Solution

Present the navigation on a menu that on narrow screen devices dynamically changes to a native select component.

***

This pattern is useful for designing a navigation menu with numerous and lengthy items that needs to work simultaneously on the mobile and desktop version. In the desktop version of the website the menu is presented expanded, on a narrow screen it is converted through JavaScript to the native select component. Thus, this pattern is normally seen in websites that implement a responsive design.

This type of menu can be a practical alternative for when vertical space is scarce and you want to display the menu on top of the page. However, it is not the most elegant of the alternatives, because it adds another layer of information with a distinctive interface. A more clean and elegant approach in terms of visual design can be achieved through the TOGGLE MENU pattern, which uses a comparable type of interaction but with a custom interface.

This type of menu is easier to recognize as something selectable because it uses the native controls of each device. Likewise, because it uses controls that are optimized for the respective device, you can be confident that it will work and be accessible in most of them. Though, as a downside, because it uses the native browser components, it is very difficult to achieve a consistent look across platforms — Figure 18.

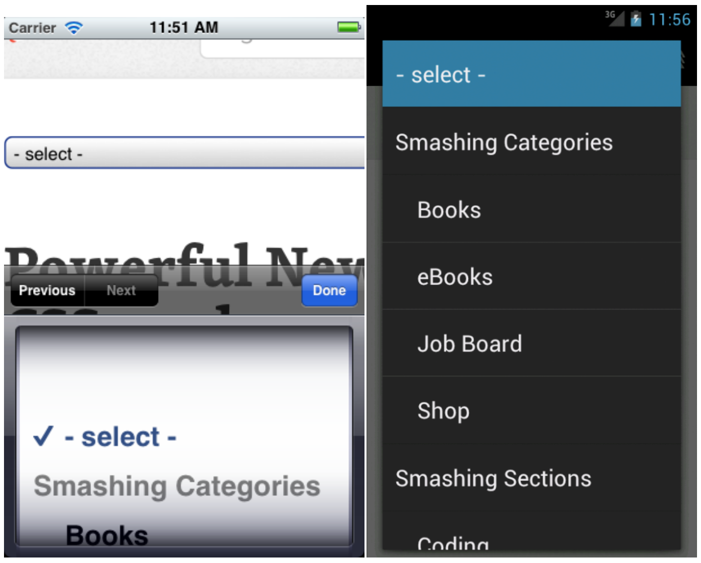
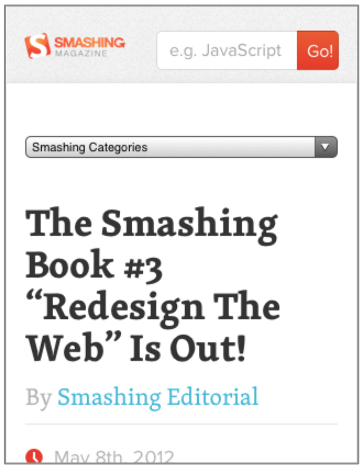


Figure 20 Screenshots from the same SELECT MENU at *Smashing Magazine* website on different platforms. iOS on the left, Android on the right.

You can also work with subitems, though, that can be even more confusing. In Figure 20 subitems are denoted by an indent, but dashes are also a common alternative.

Examples



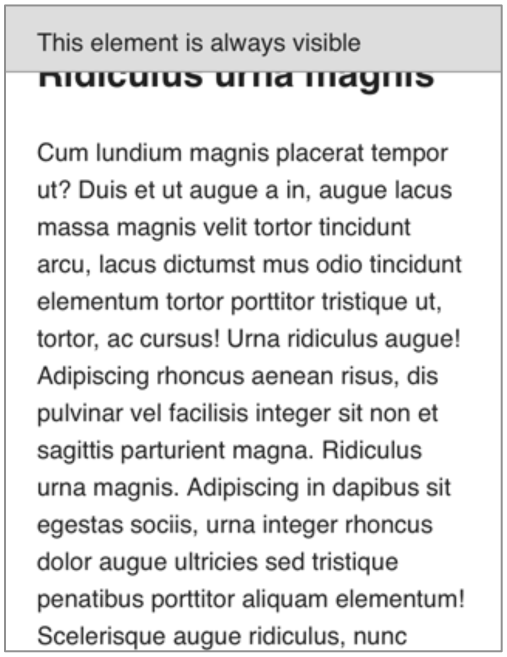
http://www.smashingmagazine.com

http://www.lancs.ac.uk
http://www.worldskillslondon2011.com
http://www.sony.com

Related Patterns

– TOGGLE MENU

4.12 Fixed Content



Cum lundium magnis placerat tempor ut? Duis et ut augue a in, augue lacus massa magnis velit tortor tincidunt arcu, lacus dictumst mus odio tincidunt elementum tortor porttitor tristique ut, tortor, ac cursus! Urna ridiculus augue! Adipiscing rhoncus aenean risus, dis pulvinar vel facilisis integer sit non et sagittis parturient magna. Ridiculus urna magnis. Adipiscing in dapibus sit egestas sociis, urna integer rhoncus dolor augue ultricies sed tristique penatibus porttitor aliquam elementum! Scelerisque augue ridiculus, nunc

Figure 21. FIXED CONTENT.

Problem

The normal behavior of content on a page is to go off the viewport as the user scrolls through. However, you may need to provide quick access to functions or information persistently through the entire page.

Solution

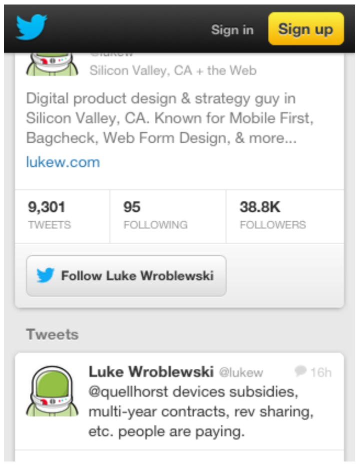
Present the content positioned fixed to the edges of the browser window, over the page, and assuring that it is visible through the entire page.

***

FIXED CONTENT allows us to provide quick access to functions that need to be present through the entire page, or alert the user of some important information. Given that, it is commonly used for designing web applications or to give a more native look to the interface.

A major drawback with this pattern is that it takes a considerable amount of space, which is already a limited asset on these devices. Besides of the already small height of the device, we have to account for the OS toolbar, the browser chrome, and a potential keyboard, all those contributing for reducing the effective the real estate of the page. This problem is even more prevalent when the device is oriented in landscape. Therefore, make sure that any FIXED CONTENT is absolutely essential in your website.

Examples



Silicon Valley, CA + the Web

Digital product design & strategy guy in Silicon Valley, CA. Known for Mobile First, Bagcheck, Web Form Design, & more...
lukew.com

9,301	95	38.8K
TWEETS	FOLLOWING	FOLLOWERS

Luke Wroblewski @lukew 16h
@quellhorst devices subsidies, multi-year contracts, rev sharing, etc. people are paying.

http://mobile.twitter.com

http://m.paper.li
http://designmadeingermany.de/magazin/5
http://2011.dconstruct.org

Related Patterns

– *Bottom Navigation*, in *Designing Interfaces* (Tidwell 2011)
– *Fixed Menu*, in *Designing Mobile Interfaces* (Hoober and Berkman 2011)

4.13 Slideshow

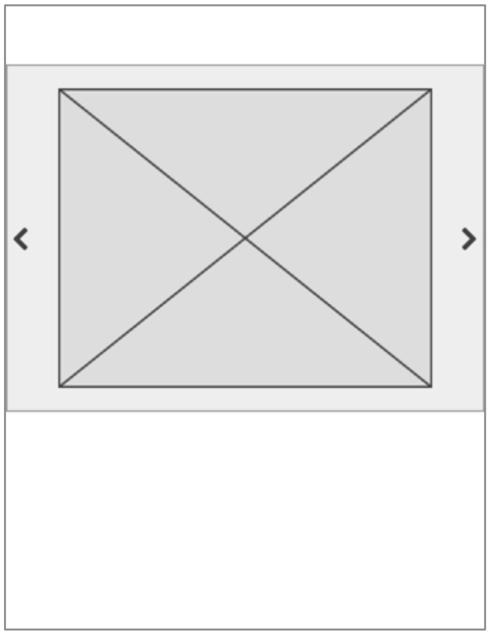


Figure 22. SLIDESHOW.

Problem

You need to display a series of related information, with comparable content in terms of length, without consuming a considerable amount of vertical space.

Solution

Reveal items one at a time, by changing the visible item automatically in a specific time interval or by providing a method for browsing through the entire content.

***

When you have a series of related content that is extensive but not considerably important, you can save vertical space by having all pieces of that content arranged horizontally, placed virtually beyond the width of the device, while keeping a window — generally with the same width of the device — that works as a viewfinder for the list. Users can only see one item of the list at each time but have some method for traverse through the list. The SLIDESHOW is a common pattern on the web, and was previously formulated as a pattern for mobile in *Designing Mobile Interfaces* (Hoober and Berkman 2011). It is commonly used with images, although it can be successfully implemented with text, or image and text. Each slide can be simply used to display information but can also work as a link.

However, do not use this pattern for presenting critical information. Since only one item will be visible at any time and users may not understand that they can scroll through the list, hidden content can easily go unnoticed. The SLIDESHOW is most suitable for presenting more casual information like a gallery of images.

Although you can use slides with different content in terms of length, it is a good idea to keep the slide height constant across slides to prevent the layout from moving up and down between slides.

Interaction Details

Signalize that additional content is hidden, moreover, use that sign as a hint of how to reveal it. An arrow or an index of the list — Figure 23 — are commonly used to solve this problem. Alternatively, you can display a portion of the previous and following images to alert users that more content is available — closer to the behavior of what is usually described as a carousel.

For scrolling through the content you can use one of these approaches or a combination of them:

- Slides change automatically without any control of the user.
- A tap on the slide to make it move to the next one; if this is the only method for navigating the SLIDESHOW, it has the inconvenient that if you have a long list and want to go back one slide, you need to scroll through the entire list.
- You can use a "next" and "previous" buttons for scrolling through the slideshow; arrows or textual descriptions are usually used for this.
- Use a swipe gesture, which is probably the most elegant and a natural of the alternatives because you are directly manipulating the content; however, because of its relative novelty it is harder to be discovered and more difficult to implement. Whenever possible try to take advantage of the swipe gesture to traverse trough the gallery, but it is a good practice to provide a fallback — a button — for devices that may not support gestures; and for users who may not understand it or are not used to this kind of interaction.

Nevertheless, implement an animation between each slide to help users grasp what is happening. A crossfade is common with this type of pattern, but an animation of the section sliding in and out can provide a better affordance, particularly if a swipe is used to move between slides.

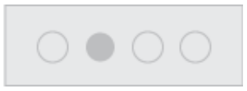
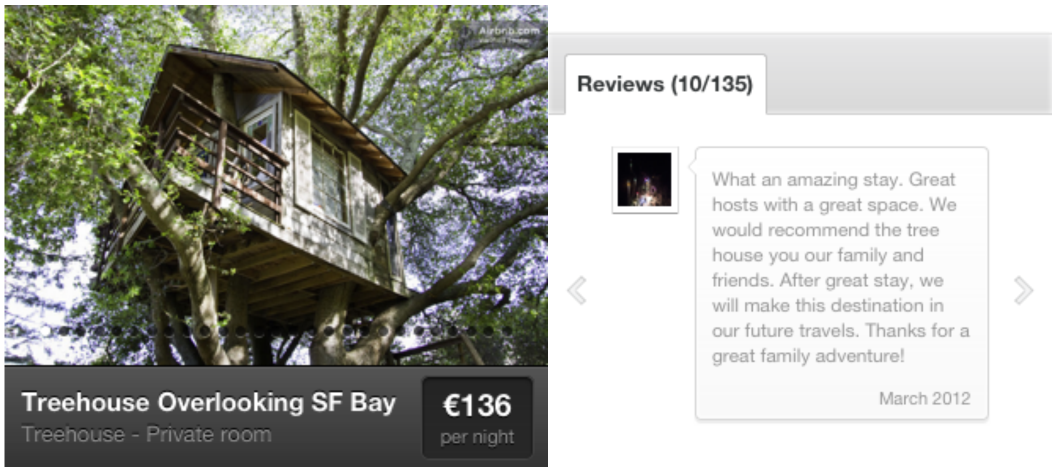


Figure 23. Index of the SLIDESHOW.

Provide some feedback of the user's position within the list. It can be done by using an index of the list — Figure 23—, and if it is important to identity each slide, you can number them.

Markers can work as buttons that link to the corresponding slide, although they need to be large enough to work efficiently. Thus, you should always provide an alternative method for scrolling through the list.



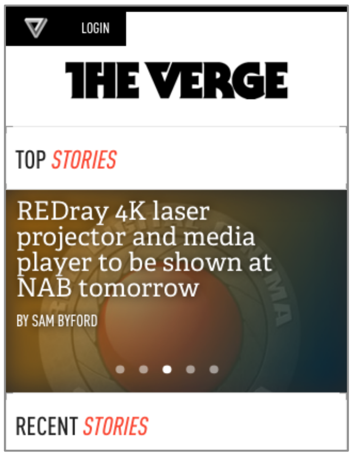
What an amazing stay. Great hosts with a great space. We would recommend the tree house you our family and friends. After great stay, we will make this destination in our future travels. Thanks for a great family adventure!

March 2012

Figure 24. *Airbnb's* product page.

The Airbnb website has two SLIDESHOW galleries in the same page, and each one implements a different method for browsing the gallery: one only works with a swipe — Figure 24, on the left — and one only works with buttons — Figure 24, on the right. Although they are used for different purposes and are visually distinct, it can still confuse users and it would be expectable that at least the interaction worked identically.

Examples


LOGIN
THE VERGE
TOP STORIES
REDray 4K laser projector and media player to be shown at NAB tomorrow
BY SAM BYFORD
RECENT STORIES

http://theverge.com

http://bostonglobe.com
http://starbucks.com
http://m.zappos.com
http://m.timhortons.com
http://www.starbucks.com
http://m.brancottestate.com
http://m.airbnb.com
http://m.nfl.com
http://etsy.com
http://unicef.se
http://mobile.engadget.com
http://m.microsoft.com

Related Patterns

– *Slideshow, Designing Mobile Interfaces* (Hoober and Berkman 2011)

4.14 Tabs

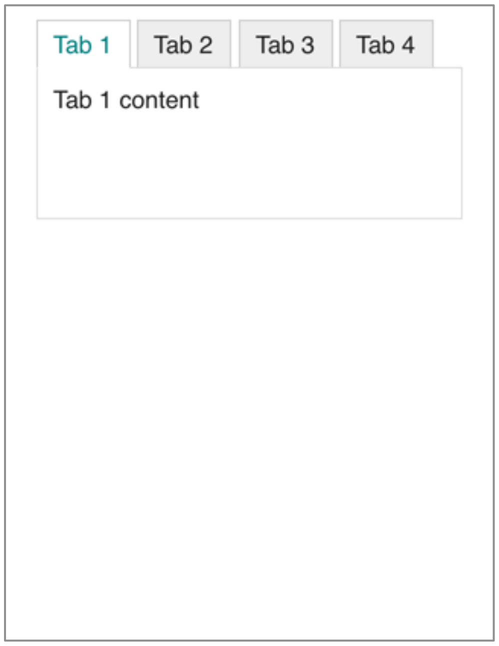


Figure 25. TABS.

Problem

You have a series of related information with a clearly defined hierarchy, and comparable length that needs to be presented at a similar level without wasting too much vertical space.

Solution

Arrange horizontally the headings of all items in the list, but display only the content of one. The visibility of each module can be toggled by users.

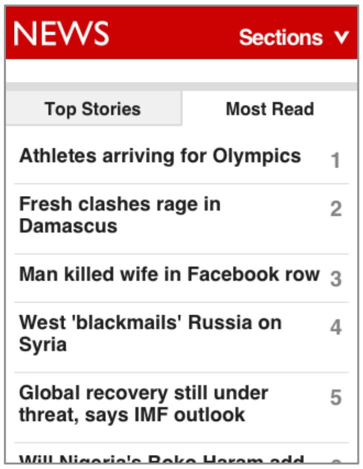
***

TABS are widely used in web design and are therefore recognizable by users. They use a metaphor of the labels on folder archives which make them easy to understand. TABS should be used to alternate between views within the same context (Nielsen 2007) rather than between pages. That is, they do not take the user to another page, only the visibility of the content changes.

You should clearly identify which is the active tab. The tab heading should be visually connected to the content for better making this distinction. You should take special attention when there are only two tabs, because the inactive state can be more easily mistaken as being active.

TABS work better when you only have a few tab modules that fit on the width of the page, and there is only one row of tabs. Two rows or more of tabs are a quite confusing and not very elegant. If you have a list of tabs headings that do not fit the width of the device it is better to truncate part of the list and to provide a method for scrolling through the tabs headings — a behavior similar to a SLIDESHOW. Nonetheless, it may be a better idea to revise your design and think of an alternative approach. A VERTICAL LIST or a SLIDESHOW can sometimes be an option.

Examples


NEWS
Sections
Top Stories
Most Read
Athletes arriving for Olympics 1
Fresh clashes rage in Damascus 2
Man killed wife in Facebook row 3
West 'blackmails' Russia on Syria 4
Global recovery still under threat, says IMF outlook 5

http://m.bbc.co.uk/news

http://m.airbnb.com

Related Patterns

– *Tabs*, in *Designing Mobile Interfaces* (Hoober and Berkman 2011)
– *Tab Menu*, in *Mobile Design Pattern Gallery* (Neil 2012)

4.15 Dropdown

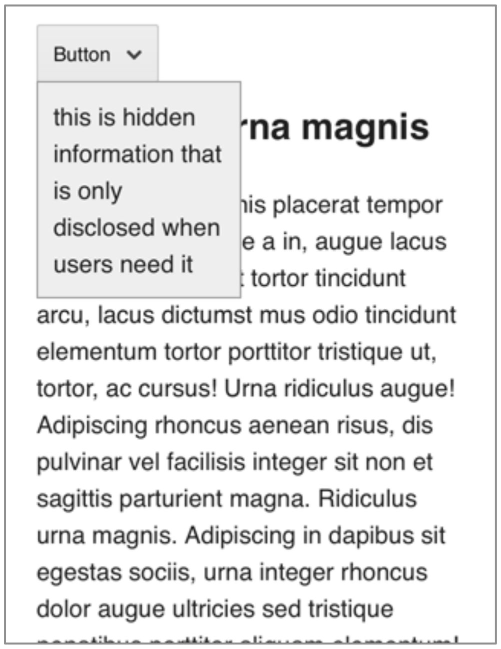

Button
this is hidden information that is only disclosed when users need it
rna magnis
his placerat tempor e a in, augue lacus t tortor tincidunt arcu, lacus dictumst mus odio tincidunt elementum tortor porttitor tristique ut, tortor, ac cursus! Urna ridiculus augue! Adipiscing rhoncus aenean risus, dis pulvinar vel facilisis integer sit non et sagittis parturient magna. Ridiculus urna magnis. Adipiscing in dapibus sit egestas sociis, urna integer rhoncus dolor augue ultricies sed tristique

Figure 26. DROPDOWN.

Problem

Sometimes you may have information that is not frequently needed. To simplify the interface it would be convenient to remove it; however, you still need to provide access to that content.

Solution

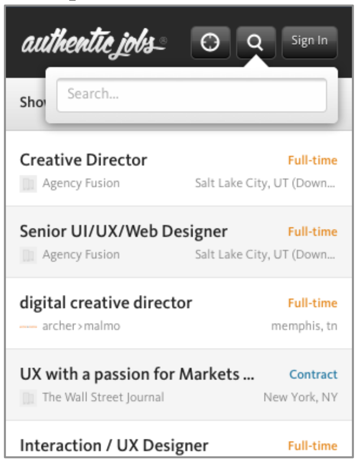
Design an element on the page that toggles the visibility of additional content. Keep the content hidden until the users express a direct intention to access it, then, make the content appear over the page.

***

You should try to not overload the page and the user with information that is not frequently needed. A DROPDOWN allows you to keep the layout simpler and cleaner by concealing non-essential information until there is a direct action of the user. You can use it to present small pieces of information that do not exceed the height of the screen. That is, you should not add additional complexity to this interface component by having the user scroll the page or the DROPDOWN to see truncated content.

In a DROPDOWN you have a button or any other element on the page that once tapped reveals the hidden content hovering on top of the page. Users should be able to withdraw it by tapping the same button again or any part of the page that is not the DROPDOWN.

Examples



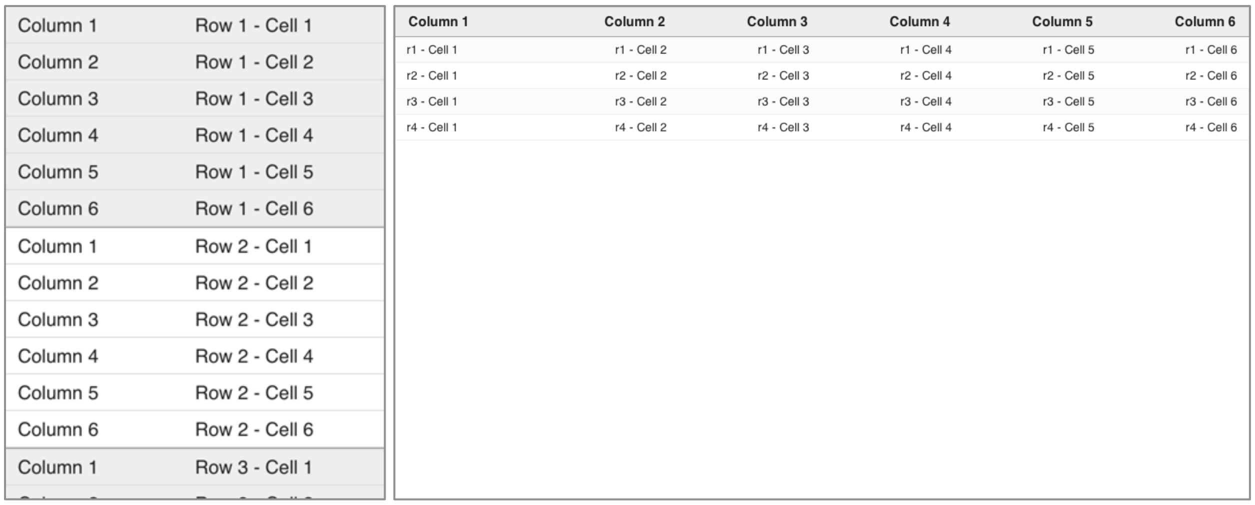
http://www.authenticjobs.com

http://facebook.com
http://m.pinterest.com
http://fringewebdevelopment.com
http://dribbble.com
http://www.london2012.com/mobile
http://m.bbc.co.uk/news
http://m.brancottestate.com
http://earthhour.fr
http://www.sony.com
Http://m.wired.com

Related Patterns

– EXPANDING LIST

4.16 Linearized Table



Column 1	Row 1 - Cell 1
Column 2	Row 1 - Cell 2
Column 3	Row 1 - Cell 3
Column 4	Row 1 - Cell 4
Column 5	Row 1 - Cell 5
Column 6	Row 1 - Cell 6
Column 1	Row 2 - Cell 1
Column 2	Row 2 - Cell 2
Column 3	Row 2 - Cell 3
Column 4	Row 2 - Cell 4
Column 5	Row 2 - Cell 5
Column 6	Row 2 - Cell 6
Column 1	Row 3 - Cell 1

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
r1 - Cell 1	r1 - Cell 2	r1 - Cell 3	r1 - Cell 4	r1 - Cell 5	r1 - Cell 6
r2 - Cell 1	r2 - Cell 2	r2 - Cell 3	r2 - Cell 4	r2 - Cell 5	r2 - Cell 6
r3 - Cell 1	r3 - Cell 2	r3 - Cell 3	r3 - Cell 4	r3 - Cell 5	r3 - Cell 6
r4 - Cell 1	r4 - Cell 2	r4 - Cell 3	r4 - Cell 4	r4 - Cell 5	r4 - Cell 6

Figure 27. LINEARIZED TABLE.

Problem

Wide tables do not fit seamlessly on small screens. If you try to design a table with a considerable number of columns you will end up with a horizontal scroll on the page.

Solution

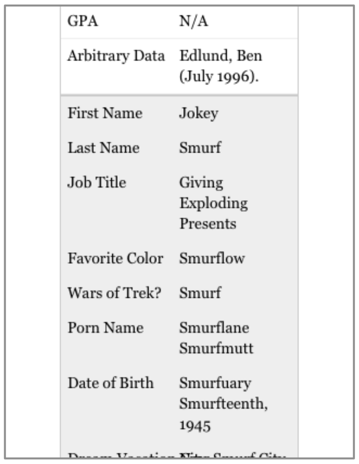
Linearize the table by converting each table row to its own table with two columns: one for the headings, another for the cells.

***

Tables can be quite wide, which is a problem on small screens. You can scale them down until they fit the screen, but that makes the text unreadable; or you can display them at normal size, but that leads to horizontal scrolling. Both alternatives are far from being desired. To overcome this problem you can reformat tables to a more linear design, in which table rows become independent entities stacked on top of each other. In this new adapted design, table headings are removed and each table row is converted to its own simplified table with only two columns: one for the table headers and another for the corresponding cells. Like in a normal table, you should also use alternated colors (or any appropriate design) in each new section so they are clearly distinguished. The idea for this pattern was proposed by Chris Coyier (2012) on the article *Responsive Data Tables*.

This approach works particularly well when you have a simple table with bi-dimensional data. With more complex tables — those that have headers with two or more levels — it can be harder to clearly linearize all the information without compromising its clarity.

Examples



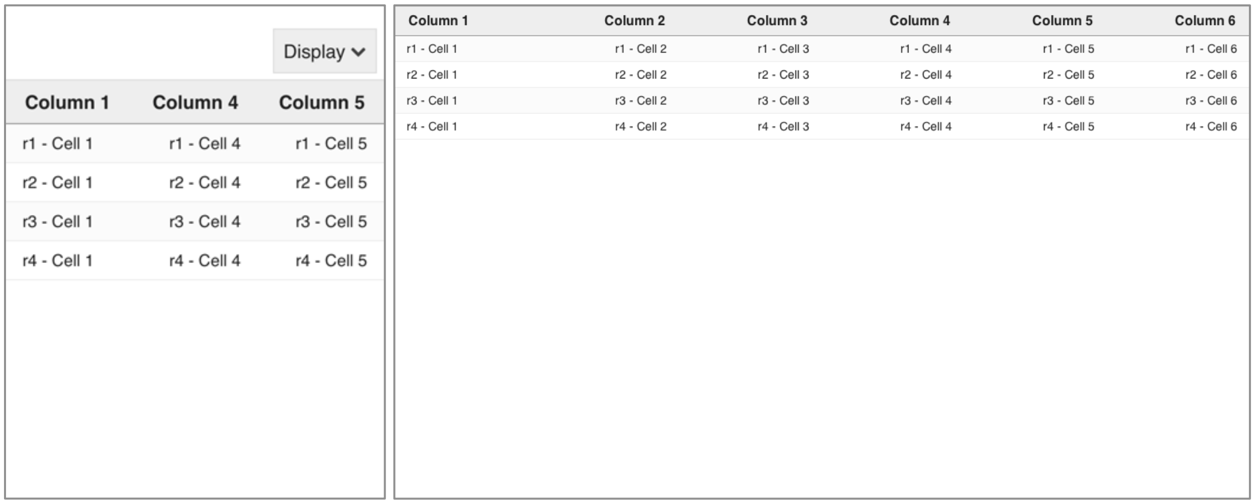
GPA	N/A
Arbitrary Data	Edlund, Ben (July 1996).
First Name	Jokey
Last Name	Smurf
Job Title	Giving Exploding Presents
Favorite Color	Smurflow
Wars of Trek?	Smurf
Porn Name	Smurflane Smurfmutt
Date of Birth	Smurfuary Smurfteenth, 1945

http://css-tricks.com/examples/ResponsiveTables/responsive.php

Related Patterns

– LINEARIZED LAYOUT
– ABRIDGED TABLE

4.17 Abridged Table



Column 1	Column 4	Column 5
r1 - Cell 1	r1 - Cell 4	r1 - Cell 5
r2 - Cell 1	r2 - Cell 4	r2 - Cell 5
r3 - Cell 1	r3 - Cell 4	r3 - Cell 5
r4 - Cell 1	r4 - Cell 4	r4 - Cell 5

Column 1	Column 2	Column 3	Column 4	Column 5	Column 6
r1 - Cell 1	r1 - Cell 2	r1 - Cell 3	r1 - Cell 4	r1 - Cell 5	r1 - Cell 6
r2 - Cell 1	r2 - Cell 2	r2 - Cell 3	r2 - Cell 4	r2 - Cell 5	r2 - Cell 6
r3 - Cell 1	r3 - Cell 2	r3 - Cell 3	r3 - Cell 4	r3 - Cell 5	r3 - Cell 6
r4 - Cell 1	r4 - Cell 2	r4 - Cell 3	r4 - Cell 4	r4 - Cell 5	r4 - Cell 6

Figure 28. ABRIDGED TABLE.

Problem

You need to present a table with several columns that does not fit on the width of a device. You could reformat the table; however, the spatial relations established on the table need to be preserved.

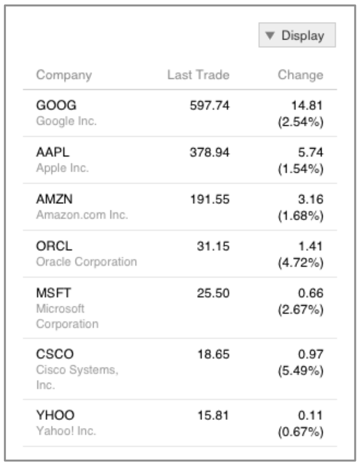
Solution

Display the table with some of the columns hidden, but provide a method for users to toggle the visibility of the hidden columns.

***

An ABRIDGED TABLE is an alternative to the LINEARIZED TABLE, particularly useful when the order and relations established on the table are important for its understanding. This pattern is most suitable for responsive designs because it allows us to automatically conceal columns on a table depending on the width of the device. It should be implemented in a way that permits to specify the order in which columns should be hidden, so non-essential columns can be removed first. You should also provide a method that allows users to reveal the hidden columns; a button that triggers a DROPDOWN with a list of all available columns can be a solution to this problem.

Examples



Company	Last Trade	Change
GOOG Google Inc.	597.74	14.81 (2.54%)
AAPL Apple Inc.	378.94	5.74 (1.54%)
AMZN Amazon.com Inc.	191.55	3.16 (1.68%)
ORCL Oracle Corporation	31.15	1.41 (4.72%)
MSFT Microsoft Corporation	25.50	0.66 (2.67%)
CSCO Cisco Systems, Inc.	18.65	0.97 (5.49%)
YHOO Yahoo! Inc.	15.81	0.11 (0.67%)

http://www.filamentgroup.com/examples/rwd-table-patterns

Related Patterns

– LINEARIZED TABLE

4.18 Touch Friendly Target

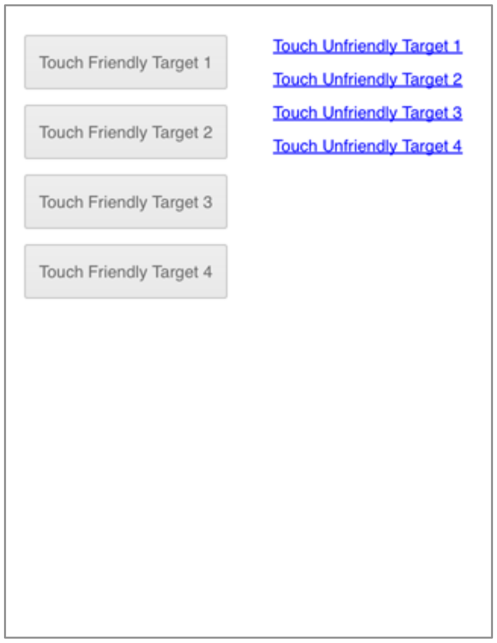

Touch Friendly Target 1
Touch Friendly Target 2
Touch Friendly Target 3
Touch Friendly Target 4
Touch Unfriendly Target 1
Touch Unfriendly Target 2
Touch Unfriendly Target 3
Touch Unfriendly Target 4

Figure 29. TOUCH FRIENDLY TARGET.

Problem

Because of the small screen, the nonexistence of tactile feedback, and the lack of precision of our *fat fingers*[7], hitting a target on a mobile device can be a challenging task. You need to design an interface that must be effortlessly used by touch.

Solution

Design all touchable elements large enough and generously spaced so they can be easily triggered.

***

With a mouse we can easily trigger very small targets, on mobile devices, because we are using our fingers as an input device that can be a more changeling task. Our fingers are much more imprecise than a mouse, as such, you should design touchable elements large enough so users can easily interact with them.

It is frustrating when we press a button and nothing happens. Currently devices provide no haptic feedback, so users cannot know for sure if they just missed the target or there is a problem with the website. You can reduce the problem of the lack of feedback by providing some visual response to the fact that a target was tapped; for example, changing the background color of the target.

In addition to larger targets, you need to account for the space between targets. You should have generous space between elements to minimize errors.

If the implementation of this pattern leads to enormous targets, you can use an ICEBERG TIP instead.

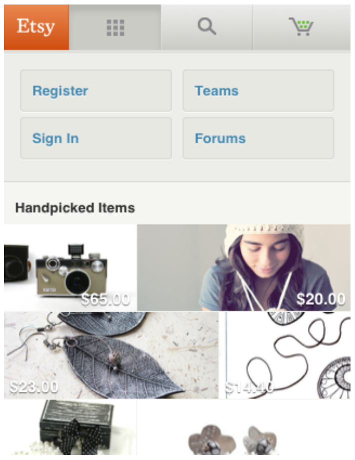
Optimal Size

[7] The 'fat finger syndrome' is a colloquial term used to describe the trigger of accidental actions caused by the fact that users' fingers are larger than the intended target.

The recommended minimum size for a target differs depending on which user interface guideline we may be following. However, the optimal size should be approximately that of an adult finger, which largely have a diameter of 16mm to 20mm (Saffer 2008), but the size in pixels varies depending on pixel density:

– The iPhone Human Interface Guidelines (Apple Inc 2012), recommends a minimum of 44x44 pixels for targets. Since the release of devices with higher DPI, Apple updated that value to an abstract measure of 44x44 points.
– User Experience Design Guidelines for Windows Phone (Microsoft 2012) recommends a 9mm target as the ideal size for all devices across Microsoft platforms, and 7mm as the minimum for the height when the width of the target is larger. It also recommends 4.2mm as the minimum visual size for a touchable item; and 2mm for the space between targets.
– Nokia Developer's (Nokia 2012) resources recommends that touchable elements should be no smaller than 10x10mm. And the minimum size for target should be: 7x7mm with 1mm gaps for index finger usage; 8x8mm with 2mm gaps for thumb usage; and list type of components should have a minimum of 5 mm line spacing.

Examples



http://etsy.com

http://m.microsoft.com

Related Patterns

– ICERBEG TIP

– *Generous Borders*, in *Designing Interfaces* (Tidwell 2011)

4.19 Iceberg Tip

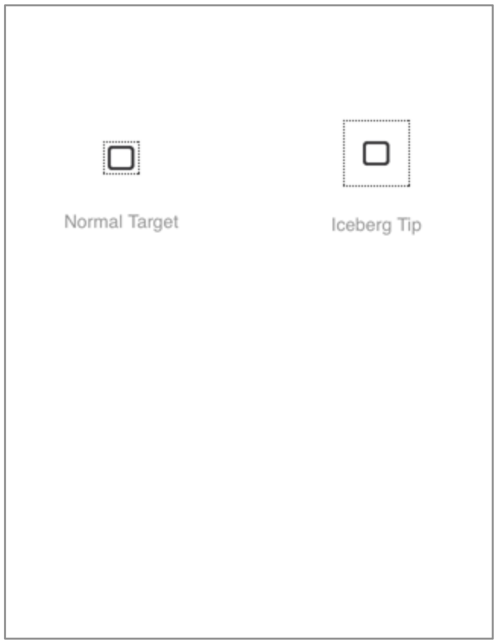


Figure 30. ICEBERG TIP.

Problem

Sometimes you need to design a TOUCH FRIENDLY TARGET, but you do not want to have enormous and inelegant buttons to clutter the interface.

Solution

Design the visible part of your object with whatever size you planned, but make the real target invisible and large enough so it can be easily touched.

***

This pattern is inspired on the idea described by Dan Saffer in *Designing Gestural Interfaces* (2008), which, like the name suggests, uses the metaphor of an iceberg for describing a target that is larger than the visible area.

When designing touch friendly interfaces, touchable elements must be large enough so users can easily interact with them. However, it is not always practical, or visually pleasing to design big buttons throughout the entire interface, in fact, touch friendly buttons may sometimes look clumsy. This pattern is valuable for when we have any element, either text or image, that by itself is not large enough to be triggered without effort; or when designing big buttons may go against what was envisioned for the visible design of the interface.

The solution involves having the visible part of the element surrounded by an invisible padding area that also works as a target, i.e., only a portion of the target is visible, the rest is hidden.

One limitation of this pattern is that it cannot be used when we have several touchable elements side by side, without creating additional white space between them.

We should always try to provide the correct affordances for the interface, but because of the small scale that the visible component can reach, the simpler design, or just because users were not expecting that elements that small would work as buttons, it may happen that these elements are not be perceived as touchable. Microsoft (2012) recognizes this downside and recommends that the graphic component should be at least 4.2mm.

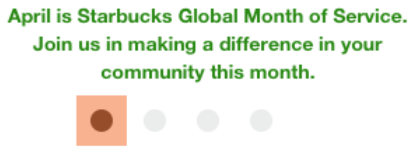

April is Starbucks Global Month of Service.
Join us in making a difference in your community this month.

Figure 31. ICEBERG TIP on Starbucks <www.starbucks.com>, the color overlay represents the real target.

Starbucks — Figure 31 — uses small circles with a diameter of 16px as buttons for navigating through a SLIDESHOW. Those buttons do not look large enough for being easily touched; however, the real target is a 36px square, which improves considerably its efficacy, though, it is still below what is usually recommended by TOUCH FRIENDLY TARGET.

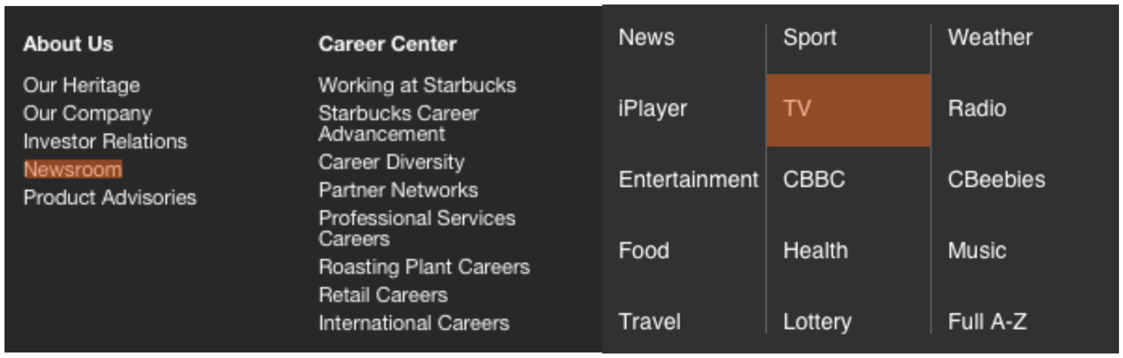
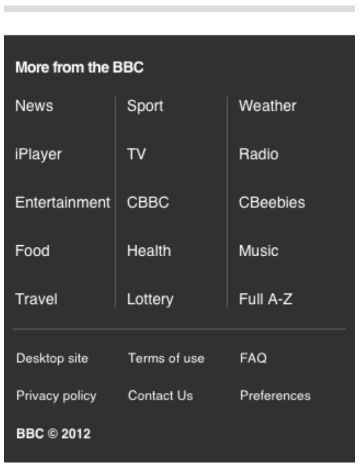


Figure 32. Comparison of targets on *Starbucks* (left) and *BBC* (right).

On the Figure 32, on the left image, we have a menu with links that could be improved by using this pattern. The text is too small, which by itself is not necessarily a problem, but the target occupies the same space as the text, in this case, as low as 8 pixels height; and there is not adequate space between links, which makes it almost impossible to hit the intended target on the first try. On the contrary, on the right image, we have a comparable menu, but here, while the visible part is still small, links have plenty of space between them and the real target occupies the maximum space possible.

Examples



http://m.bbc.co.uk/news

http://www.starbucks.com
http://www.pepatostudio.com
http://www.earthhour.fr

Related Patterns

– TOUCH FRIENDLY TARGETS

4.20 Dynamic Filtering

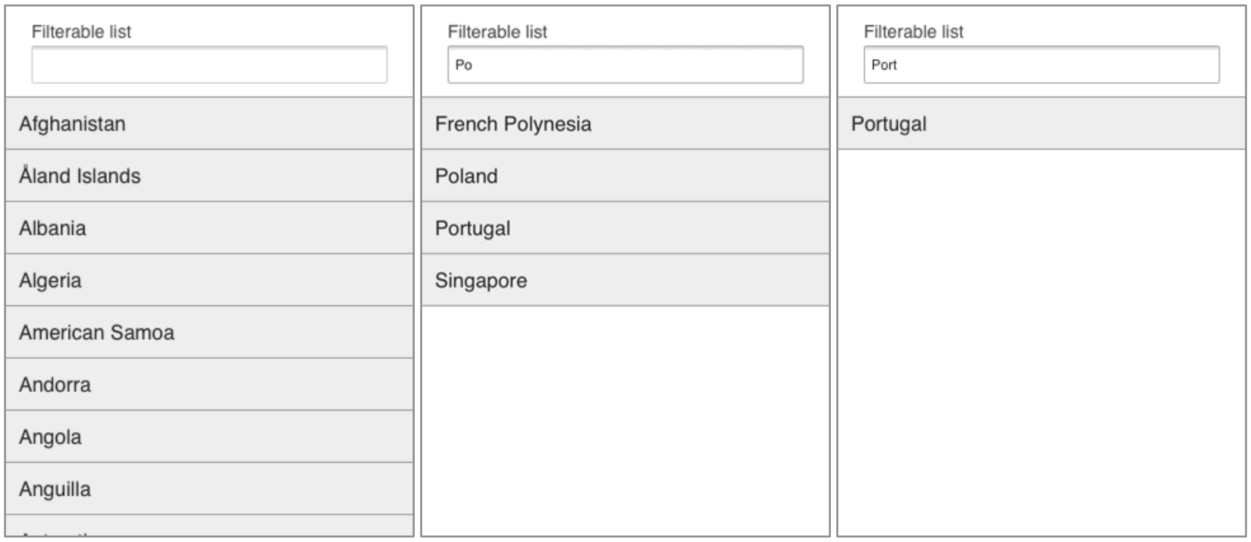


Figure 33. DYNAMIC FILTERING.

Problem

Searching through an extensive list of items within a page can be a tiresome task. Likewise, searching on a long dataset, especially if the user does not remember exactly the search term, can also be very time consuming.

Solution

Provide a search form that dynamically filters the results as the user is typing.

***

The idea for the DYNAMIC FILTERING can be found in patterns such as, *Dynamic Search* (Neil 2012) or *Search Within* (Hoober and Berkman 2011), and you can implement it on forms in order to present faster results, by minimizing the users' need to type and scroll. Unlike an explicit search that forces users to type the complete search term and press a button confirm it, with a DYNAMIC FILTERING they can get the intended result by typing only a few letters. Because users do not need to type everything, this pattern can also be helpful for cases when they do not remember accurately the desired query.

When the user types a letter the Dynamic Filtering removes entries that do not contain that letter. As the user keeps typing, the system keeps eliminating entries that do not fit the pattern entered.

Examples

http://www.google.com
http://www.bing.com

Related Patterns

– *Dynamic Search*, in *Mobile Design Pattern Gallery* (Neil 2012)
– *Search Within*, in *Designing Mobile Interfaces* (Hoober and Berkman 2011)

4.21 Clear Entries

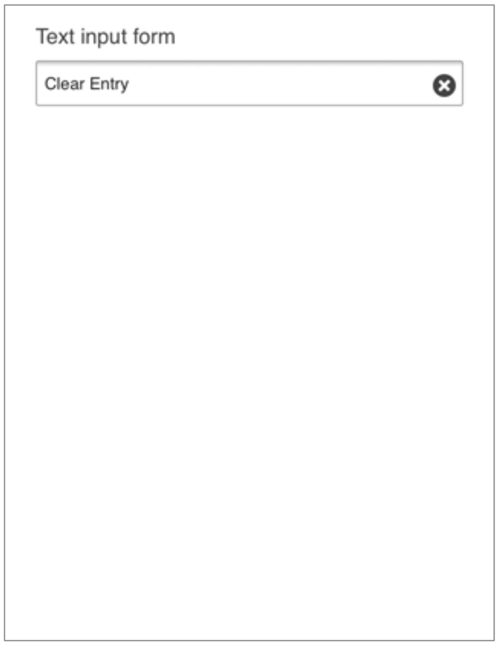


Figure 34. CLEAR ENTRIES

Problem

Although typing on a virtual keyboard is a difficult task, resetting an input form to the default state may be no less easy. Deleting a long string of text letter by letter can be a very tedious error prone task.

Solution

Provide a button that resets the input form with one tap.

***

You should always strive for minimize users' need to input text on mobile. Like text entry, deleting long strings of text can be a very tedious task and propitious to mistakes. While, generally, OS have some sort of method to facilitate the clearing of input fields, such as, faster deleting on long presses, you may still provide a better and faster method for this task. This pattern is based on the patterns *Text Clear Button* (Tidwell 2011) and *Clear Entry* (Hoober and Berkman 2011), so you can find additional information there.

Therefore, provide a button on all free-text input fields that allows users to quickly remove previous composed text. Place that button inside the input field aligned to the right and farther enough from other targets so it is not tapped by mistake. If it is needed you can use and ICEBERG TIP to improve the efficacy of that button. A button with an “x” is succinct, unambiguous and almost a standard so it is usually favored, but if you have the space you can use a label like “Clear” or “Reset”.

Examples

http://google.com
http://bing.com

Related Patterns

– *Clear Entry*, in *Designing Mobile Interfaces* (Hoober and Berkman 2011)
– *Text Clear Button*, in *Designing Interfaces* (Tidwell 2011)

REFERENCES

Alexander, Christopher, Sara Ishikawa, and Murray Silverstein. *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford University Press, 1977.
Apple Inc. "Platform Characteristics". iOS Human Interface Guidelines, 2012. 3 June 2012. <http://developer.apple.com/library/ios/#DOCUMENTATION/UserExperience/Conceptual/MobileHIG/Characteristics/Characteristics.html>.
Borchers, Jan. *A Pattern Approach to Interaction Design*. Chichester: Wiley Publishing, 2001.
Clark, Josh. *Tapworthy: Designing Great Iphone Apps*. Sebastopol, CA: O'Reilly Media, 2010.
Clarke, Andy. "We Need a Standard Show Navigation Icon for Responsive Web Design". 2012. 3 June 2012. <http://www.stuffandnonsense.co.uk/blog/about/we_need_a_standard_show_navigation_icon_for_responsive_web_design>.
Coyier, Chris. “Responsive Data Tables.” CSS Tricks, 2011. 10 July 2012. <http://css-tricks.com/responsive-data-tables>.
Crumlish, Christian, and Erin Malone. *Designing Social Interfaces: Principles, Patterns, and Practices for Improving the User Experience*. Sebastopol, California: O'Reilly Media, 2009.
Pattern Factory. "Patternry Open Library". 2009. 18 January 2012. <www.patternry.com>.
Firtman, Maximiliano. *Programming the Mobile Web*. Sebastopol, California: O'Reilly, 2010.
Frost, Brad. “Responsive Navigation Patterns.” Brad Frost Web, 2012. 3 June 2012. <wwww.bradfrostweb.com/blog/web/responsive-nav-patterns>.
Gamma, Erich, et al. *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley Professional, 1995.
Hoober, Steven, and Eric Berkman. *Designing Mobile Interfaces*. Sebastopol, California: O'Reilly Media, 2011.
Microsoft. "Interactions and Usability with Windows Phone". User Experience Design Guidelines for Windows Phone, 2012. 3 June 2012. <http://msdn.microsoft.com/en-us/library/hh202889(v=VS.92).aspx>.
Neil, Theresa. *Mobile Design Pattern Gallery*. Sebastopol, California: O'Riley, 2012.
Nielsen, Jakob. "Tabs, Used Right". 2007. Use It. 3 June 2012. <http://www.useit.com/alertbox/tabs.html>.
Nokia. "Scale and Positioning of Controls". Nokia Developer, 2012. 3 June 2012. <http://library.developer.nokia.com/index.jsp?topic=/S60_5th_Edition_Cpp_Developers_Library/GUID-5486EFD3-4660-4C19-A007-286DE48F6EEF.html>.
Saffer, Dan. *Designing Gestural Interfaces: Touchscreens and Interactive Devices*. Sebastopol, California: O'Reilly Media, 2008.
Schümmer, Till, and Stephan Lukosch. *Patterns for Computer-Mediated Interaction.* Chichester, England: Wiley Publishing, 2007.
Scott, Bill, and Theresa Neil. *Designing Web Interfaces: Principles and Patterns for Rich Interactions.* Sebastopol, California: O'Reilly Media, 2009.
Tidwell, Jenifer. "Common Ground: A Pattern Language for Human-Computer Interface Design." 1999. Print.
———. *Designing Interfaces*. 2006. 2nd ed. Berkeley, California: O'Reilly Media, 2011.
Toxboe, Anders. "Ui Patterns". 18 January 2012. <www.ui-patterns.com>.
van Duyne, Douglas K., James A. Landay, and Jason I. Hong. *The Design of Sites: Patterns for Creating Winning Websites.* 2nd ed. Upper Saddle River, New Jersey: Prentice Hall, 2006.
Wroblewski, Luke. *Mobile First*. New York: A Book Apart, 2011.
Yahoo! "Yahoo! Design Pattern Library". 2006. 18 January 2012. <http://developer.yahoo.com/ypatterns>