Getting Around: Navigation, Signposts, and Wayfinding

- The patterns in this chapter deal with the problem of navigation.
- How do users know where they are now, where to go next, and how to get there from here?
- Navigation is a “problem” because navigating around a website or application is like commuting. You have to do it to get where you need to go, but it’s dull, it’s sometimes infuriating, and the time and energy you spend on it just seems wasted.

- The best kind of commuting is none at all. Having everything you need right at your fingertips without having to travel somewhere is pretty convenient. Likewise, keeping most tools “within reach” on an interface is handy, especially for intermediate-to-expert users (i.e., people who have already learned where everything is).

Staying Found

- Let's say you've built a large website or application that you've had to break up into sections, subsections, specialized tools, pages, windows, wizards, and so forth. How do you help users navigate?

- ***Signposts*** are features that help users figure out their immediate surroundings. Common signposts include page and window titles, web page logos and other branding devices, tabs, and selection indicators.

- ***Wayfinding*** is what people do as they find their way toward their goal. The term is pretty self-explanatory. But how people actually do it is quite a research subject—specialists from cognitive science, environmental design, and website design have studied it.

Staying Found

- Let's say you've built a large website or application that you've had to break up into sections, subsections, specialized tools, pages, windows, wizards, and so forth. How do you help users navigate?

- ***Signposts*** are features that help users figure out their immediate surroundings. Common signposts include page and window titles, web page logos and other branding devices, tabs, and selection indicators.

- ***Wayfinding*** is what people do as they find their way toward their goal. The term is pretty self-explanatory. But how people actually do it is quite a research subject—specialists from cognitive science, environmental design, and website design have studied it.

Staying Found

- These common-sense features help users with wayfinding:

Good signage

  - Clear, unambiguous labels anticipate what you're looking for and tell you where to go; signs are where you expect them to be, and you're never left standing at a decision point without guidance.
  - You can check this by walking through the artifact you're designing and following the paths of all the major use cases.
  - Make sure that each point where a user must decide where to go next is signed or labeled appropriately. Use strong "calls to action" on the first pages that a user sees.

Staying Found

Environmental clues

- You'd look for restrooms in the back of a restaurant, for instance, or a gate where a walkway intersects a fence.
- Likewise, you would look for an "X" close button at the top right of a modal dialog and logos in the upper-left corner of a web page.
- Keep in mind that these clues are often culturally determined, and someone new to the culture (e.g., someone who's never used a given operating system before) will not be aware of them.

Staying Found

Maps

- Sometimes people go from sign to sign or link to link without ever really knowing where they're going in a larger frame of reference. (If you've ever found your way through a strange airport, that's probably what you did.)

- But some people might prefer to have a mental picture of the whole space, especially if they're there often. Also, in badly signed or densely built spaces, such as urban neighborhoods, maps may be the only navigational aids people have.

The Cost of Navigation

- When you walk into an unfamiliar room, you look around. In a fraction of a second, you take in the shape of the room, the furnishings, the light, the ways out, and other clues; very quickly, you make some assumptions about what this room is and how it relates to why you walked in.

- Then you need to do what you came in to do. Where? How? You might be able to answer immediately—or not. Or maybe you're just distracted by other interesting things in the room.

- Similarly, bringing up a web page or opening a window incurs a cognitive cost. Again, you need to figure out this new space: you take in its shape, its layout, its contents, its exits, and how to do what you came to do. All of this takes energy and time. The "context switch" forces you to refocus your attention and adjust to your new surroundings.

The Cost of Navigation

- Even if you're already familiar with the window (or room) you just went into, it still incurs a cost. Not a large cost, but it adds up—especially when you figure in the actual time it takes to display a window or load a page. This is true whether you're dealing with web pages, application windows, dialog boxes, or device screens.

- The decisions that users make about where to go are similar—labels still need to be read or icons decoded, and the users will still make leaps of faith by clicking on links or buttons they're not sure about.

- Furthermore, loading time affects people's decisions. If a user clicks through to a page that takes too long to load—or fails to load altogether—he may be discouraged, and may just close the page before he finds what he came for. (So, how many viewers is that sidebar video player costing you?) Also, if a site's pages take a chronically long time to load, users will be less likely to explore that site.

Keep Distances Short

- Knowing that there's a cost associated with jumping from page to page, you can understand now why it's important to keep the number of those jumps down. When a common task requires many page jumps, try to reduce it to one or two.

- But the real efficiency gains come from the structure of the application. One of the nastiest things a designer can do is force a user to go into multiple levels of subpages, dialogs, and so forth every time he needs to accomplish a simple and everyday task. (Worse is to lead him there, tell him he can't accomplish it because of some missing precondition, and send him back to square one.).

- Can you design your application so that the most common 80% of use cases can be done in one page, without any context switches? (Or perhaps only one?)

Navigational Models

What is the navigational model for your site or app? In other words, how do the different screens (or pages, or spaces) link to each other, and how do users move between them?

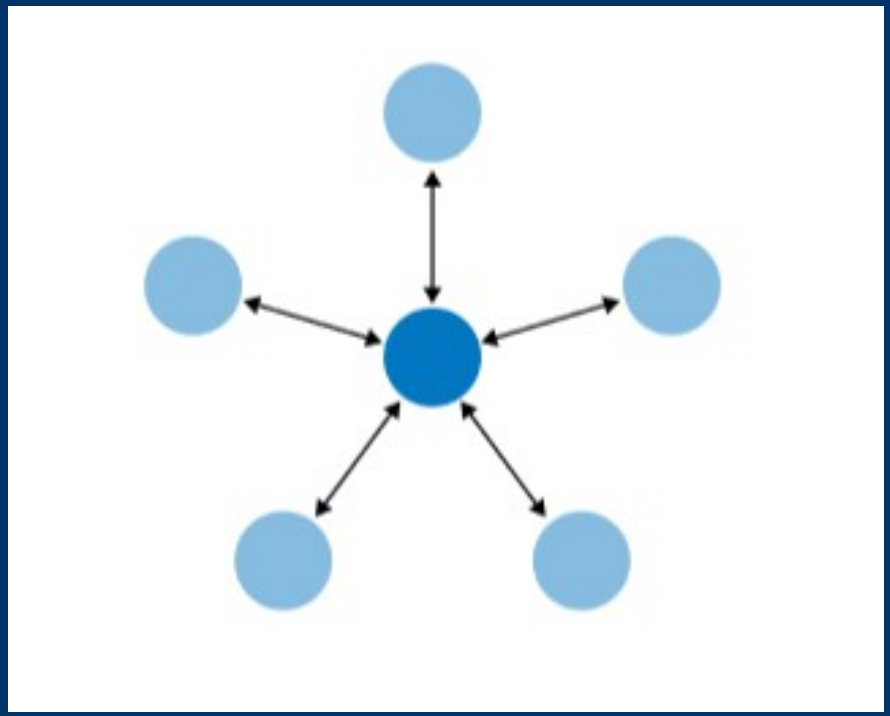
- ***Global navigation*** is what's found on every main screen. It usually takes the form of menus, tabs, and/or sidebars, and this is how users move around the formal navigational structure of the site.
- ***Utility navigation***, also found on every main screen, contains links and tools related to noncontent aspects of the site or application: sign-in, help, print, Settings Editors, language tools, and so on.
- ***Associative*** and inline navigation embed links in or near the actual content. As the user reads or interacts with the site, these links present options that might be immediately relevant to the user. They tie content together thematically.

Navigational Models

Hub and spoke

- Most often found on mobile devices, this architecture (Figure 3-1) lists all the major parts of the site or app on the home screen, or “hub.” The user clicks or taps through to them, does what she needs to do, and comes back to the hub to go somewhere else.
- The “spoke” screens focus tightly on their jobs, making careful use of space—they may not have room to list all the other major screens. The iPhone home screen is a good example; the Menu Page pattern found on some websites is another.

Hub and spoke

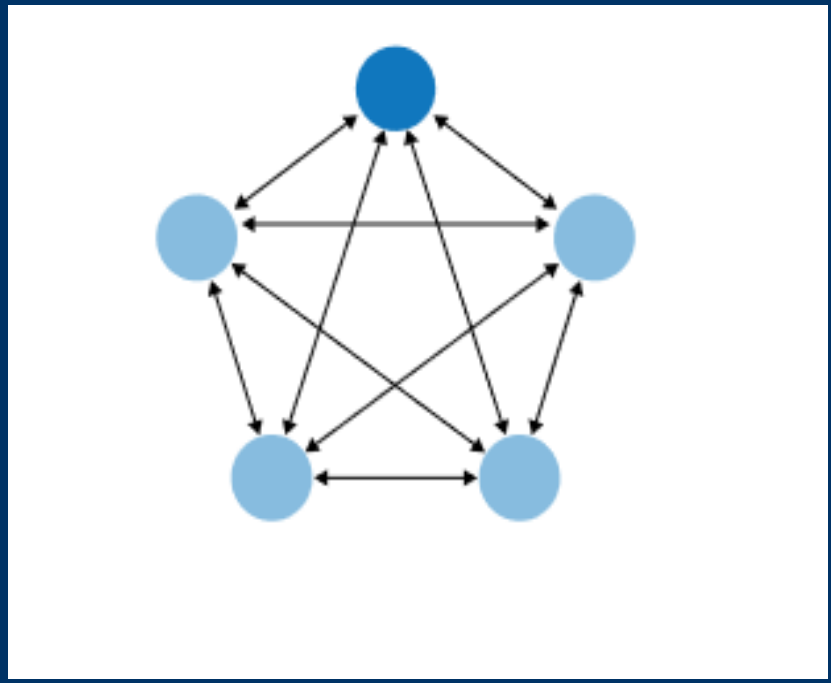


Navigational Models

Fully connected

- Many websites follow this model. There's a home page or screen, but it and every other page link to all the others—they each have a global navigation feature, such as a top menu. The global navigation may be a single level (with only five pages), or it might be deep and complex, with multiple levels and deeply buried content.
- As long as the user can reach any page from any other with a single jump, it's fully connected.

Fully connected

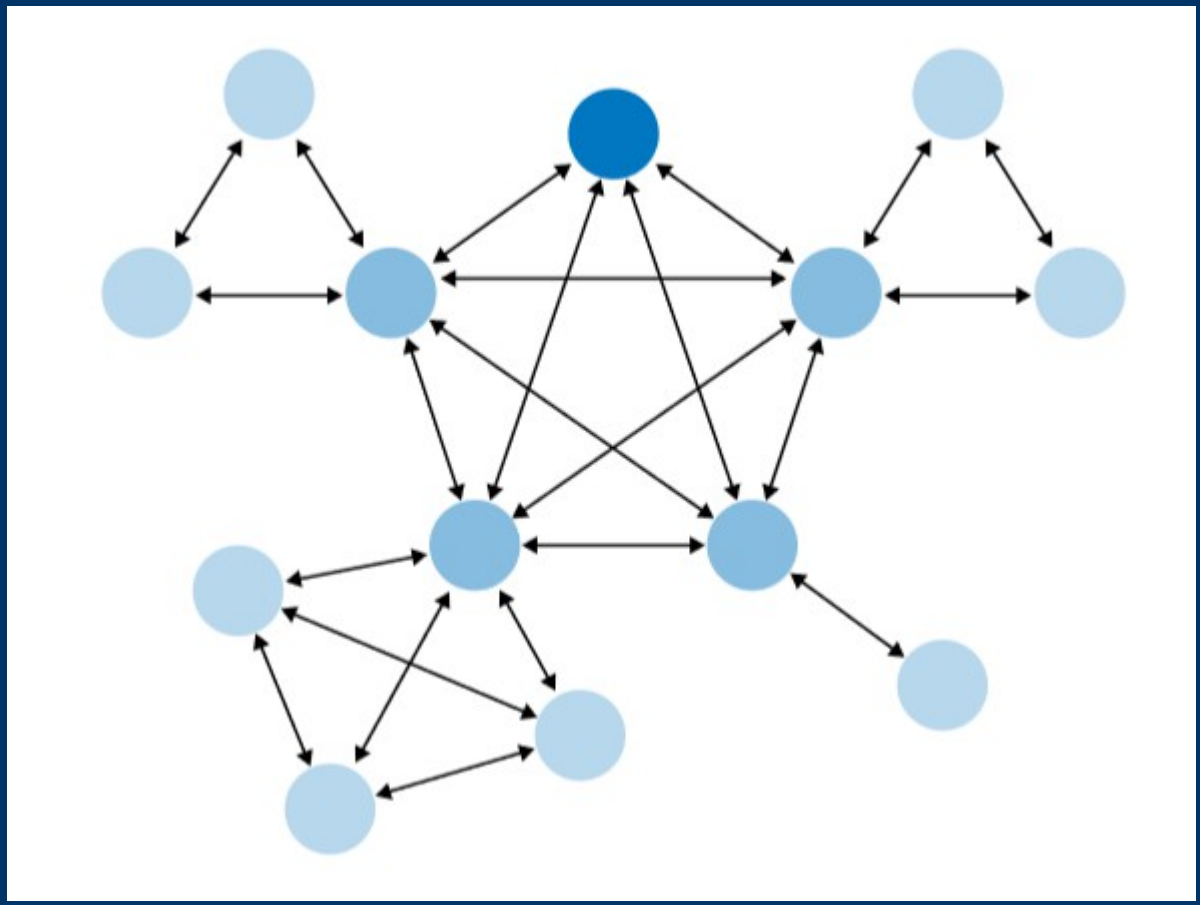


Navigational Models

Multi-level

- This is also common among websites. The main pages are fully connected with each other, but the subpages are only connected among themselves (and usually to the other main pages, via global navigation).
- You've seen this on sites that have subpages listed only in sidebars or subtabs—users see these on menus that only show up after they've clicked the link for the main page or category. It takes two or more jumps to get from one arbitrary subpage to another.
- Using drop-down menus, the Fat Menus pattern, or the Sitemap Footer pattern with a multi-level site converts it to a fully connected one, which is preferable.

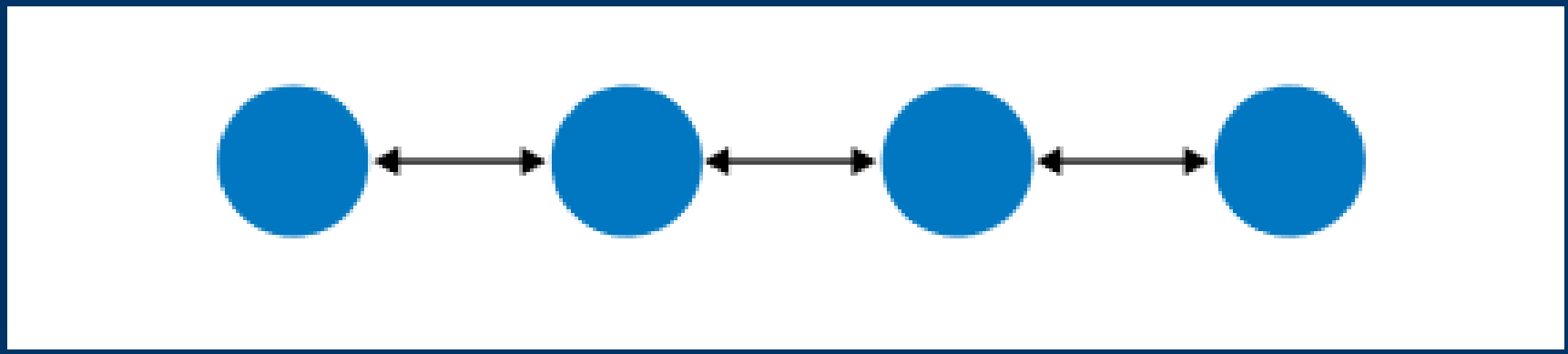
Multi-level



Navigational Models

Stepwise

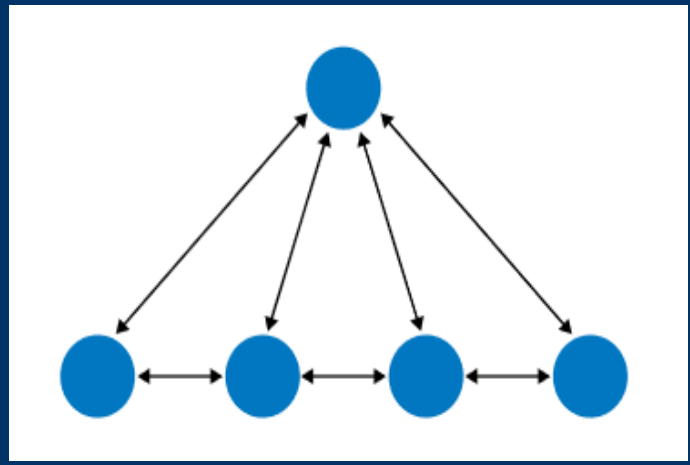
- Slideshows, process flows, and Wizards (see Chapter 2) lead the user step by step through the screens in a prescribed sequence. Back/Next links are prominent on the page.



Navigational Models

Pyramid

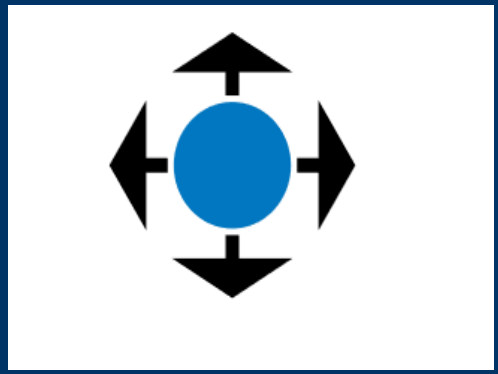
- A variant on the stepwise model, a pyramid uses a hub page or menu page to list an entire sequence of items or subpages in one place. The user picks out any item, jumps to it, and then has the option to use Back/Next links to step through other items in order. He can go back to the hub page anytime.



Navigational Models

Pan-and-zoom

- Some artifacts are best represented as single large spaces, not many small ones. Maps, large images, large text documents, information graphics, and representations of timebased media (such as sound and video) fall into this category.
- Panning and zooming are still navigation—so offer controls for panning (moving horizontally or vertically), zooming in and out, and resetting to a known position and state.



Navigational Models

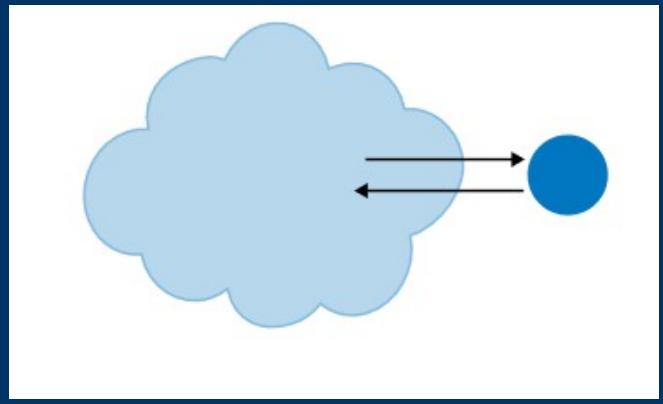
Flat navigation

- Some types of applications need little or no navigation at all.
- Consider Canvas Plus Palette applications such as Photoshop, or other complex apps such as Excel—these offer tons of tools and functions that are easily reached via menus, toolbars, and palettes.
- Tools that don’t act immediately upon the work may be accessible via Modal Panels or step-by-step progressions.
- These types of applications seem to be qualitatively different from the other navigation styles listed here: the user always knows where he is, but he may not easily find the tools he needs because of the sheer number of features available at one time.

Navigational Models

Modal panel

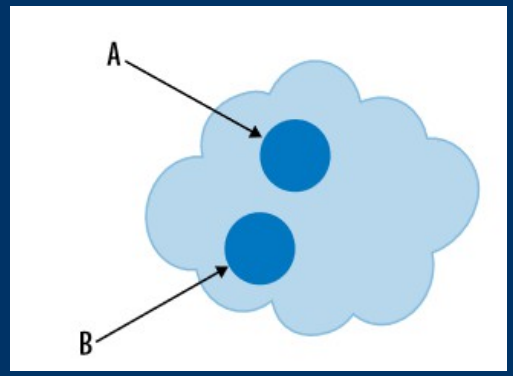
- This brings a user to a screen with no navigation options other than acknowledging its message, completing its form, or clicking the panel away.
- Modal panels often show up layered on top of a full screen or page, and are used for small, focused tasks that require the user's full attention. See the Modal Panel pattern for more discussion.



Navigational Models

Clear entry points

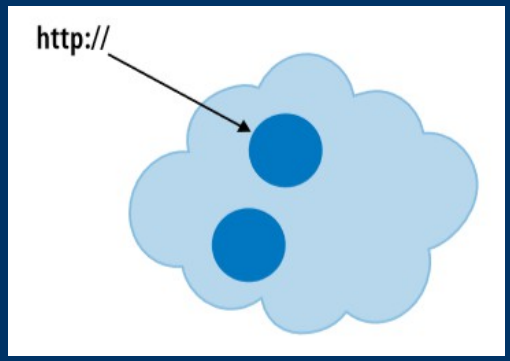
- How does a user know where to start in a complex site or app?
- The Clear Entry Points pattern shows him where to go first.
- For first-time and infrequent users, it removes some of the burden of learning the site.



Navigational Models

Bookmarks

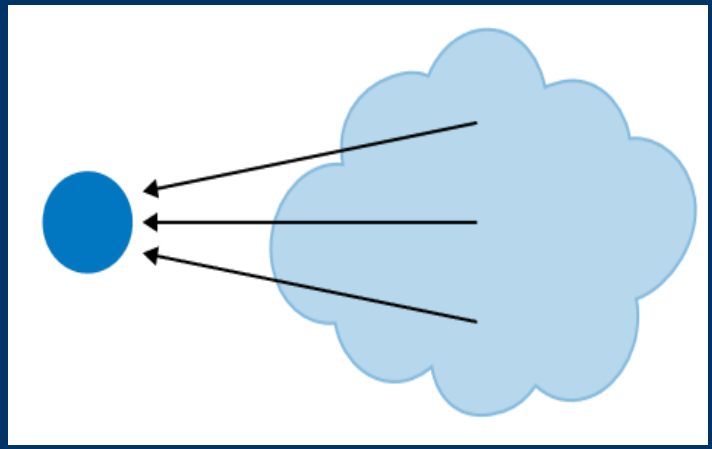
- Bookmarks, permalinks, deep links, and Deep-linked State are all ways for a user to conveniently navigate to a point of his choice, anytime he wants, even if it's deep inside a navigational structure.
- These give him a way to avoid traversing many links to get to a desired page or state.



Navigational Models

Escape hatch

- When a user is hopelessly entangled in an app, reaches an error state, or gets deeplinked into a page that he has no context for understanding, he needs an escape hatch, a well-labeled link to get back to a known place. See the Escape Hatch pattern.



Navigational Models

There are three things to notice about these navigational models.

- The first is that they're mix-and-match—an app or site might combine several of these, especially Modal Panel, Clear Entry Points, bookmarks, and Escape Hatch, which are very local and don't affect the site-wide navigation strategy.
- The second thing is that some of these mechanisms actually restrict a user's navigation options.
    - Most of the time, open access and short jumps are good things. But when a user is in the middle of a full-screen slideshow, she doesn't want to see a complicated global navigation menu! She would rather just focus on the slideshow itself, so Back/Next controls and an Escape Hatch are all that's necessary.

Navigational Models

- Third, all these mechanisms and patterns can be rendered on-screen in different ways.
  - A complex site or app might use tabs, or menus, or a sidebar tree view to show the global navigation on each page—that's something you don't need to decide until you start laying out the page.
  - Likewise, a modal panel might be done with a lightbox or an actual modal dialog—but you can postpone that until you know what needs to be modal and what doesn't.

The Patterns

1. Clear Entry Points
2. Menu Page
3. Pyramid
4. Modal Panel
5. Deep-linked State
6. Escape Hatch
7. Fat Menus
8. Sitemap Footer
9. Sign-in Tools
10. Sequence Map
11. Breadcrumbs
12. Annotated Scrollbar
13. Animated Transition

Clear Entry Points

- What:

Present only a few main entry points into the interface; make them task-oriented and descriptive. Use clear calls to action.

- Use when:

You're designing a site or app that has a lot of first-time or infrequent users. Most of these users would be best served by reading a certain piece of introductory text, doing an initial task, or choosing from a very small number of frequently used options.

However, if the purpose is clear to basically everyone who starts it, and if most users might be irritated by one more navigation step than is necessary (like applications designed for intermediate-to-expert users), this may not be the best design choice.

Clear Entry Points

- Whay:

Some applications and websites, when opened, present the user with what looks like a morass of information and structure: lots of tiled panels, unfamiliar terms and phrases, irrelevant ads, or toolbars that just sit there disabled. They don’t give the hesitant user any clear guidance on what to do first. “OK, here I am. Now what?”

For the sake of these users, list a few options for getting started. If those options match a user’s expectations, he can confidently choose one and begin working—this contributes to immediate gratification. If not, at least he knows now what the site or app actually does, because you’ve defined the important tasks or categories up front. You’ve made the application more self-explanatory.

Clear Entry Points

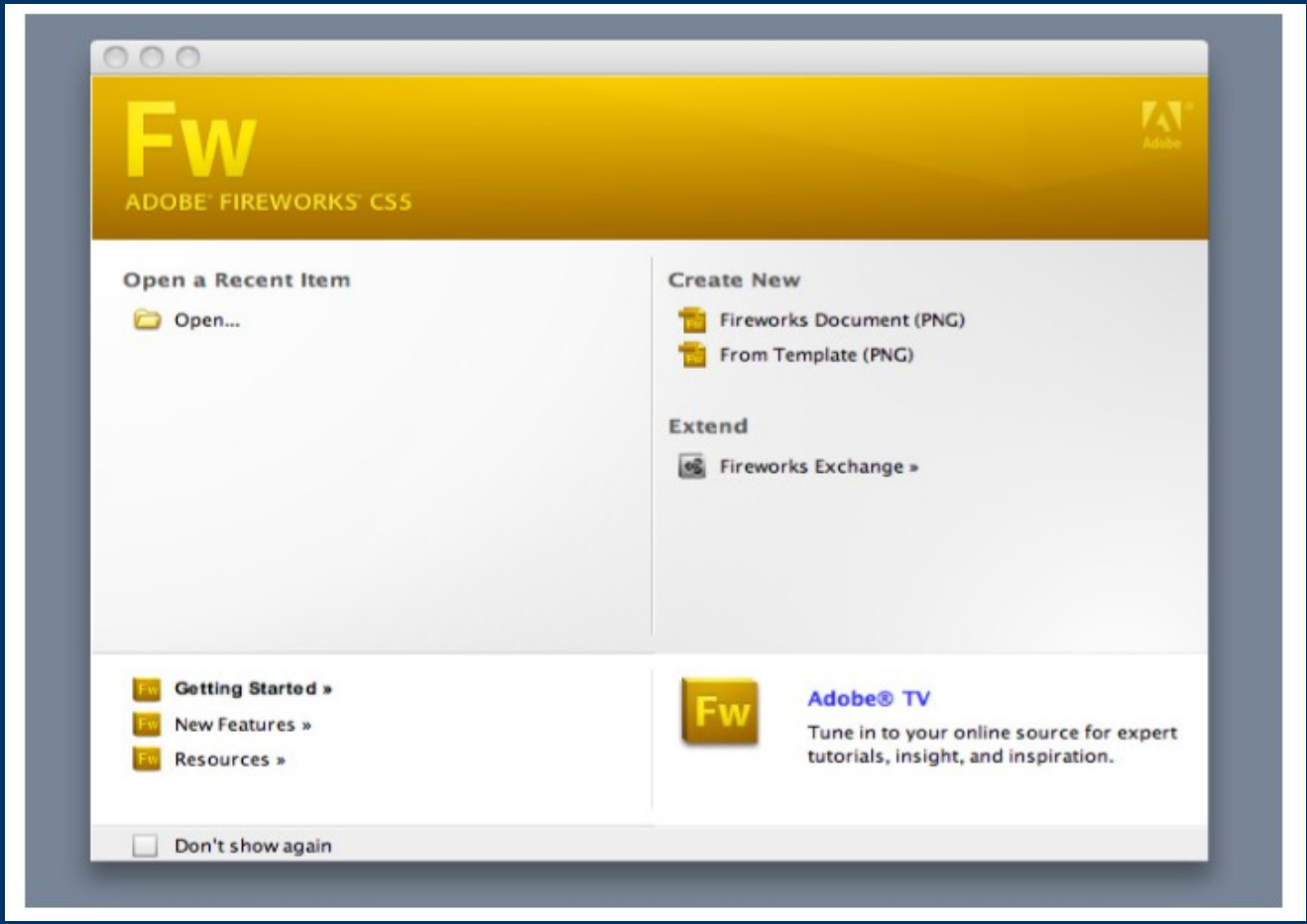
When the site is visited or the application started, present these entry points as "doors" into the main content. From these starting points, guide the user gently and unambiguously into the application until he has enough of a context to continue by himself.

Collectively, these entry points should cover most of the reasons most users would be there. There might only be one or two entry points, or many; it depends on what fits your design. But you should phrase them with language first-time users can understand—this is not the place for application-specific tool names.

Visually, you should show these entry points with emphasis proportional to their importance.

Clear Entry Points - example

Clear Entry Points - example



Menu Page

- What:

Fill the page with a list of links to content-rich pages in your site or app. Show enough information about each link to enable the user to choose well. Show no other significant content on the page.

- Use when

You're designing a home page, starting screen, or any other screen whose purpose is to be just a "table of contents"—to show where users can go from here. You may not have room for featured content (such as an article, video, or promotion), or you may simply want to let the user pick a link with no distractions.

Menu Page

Mobile apps and sites especially need Menu Pages to make the best use of their small screens.

If your (full-size) site needs to “hook” visitors into staying on the page, it may be better to use some of the page space for promotional items or other interesting content, and a Menu Page wouldn’t be the right design choice.

It takes some audacity to design a Menu Page, because you must be very confident that:

- Visitors know what the site or app is about.
- They know what they came for and how to find it.
- They wouldn’t be interested in news, updates, or features.

Menu Page

- Why:

With no distractions, users can focus all their attention on the available navigation options. You get the entire screen (or most of it, anyway) to organize, explain, and illustrate those links, and can thus direct users to the most appropriate destination page for their needs.

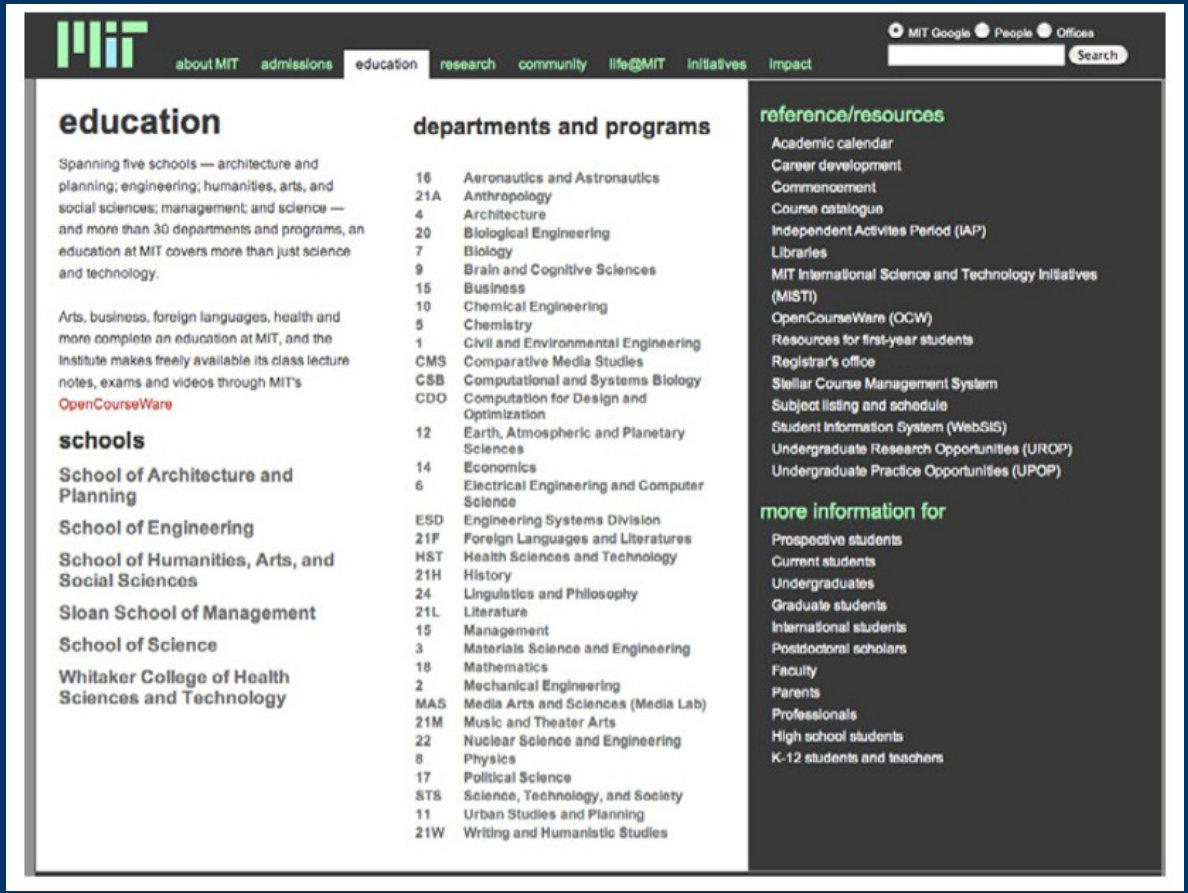
Advice:

First, label the links well, and provide just enough contextual information for users to decide where to go.

Second, consider the visual organization of the list of links. Do they come in categories, or perhaps a two- or three-level hierarchy? Is it ordered by date? Express that organizational scheme in the list.

Third, don't forget a search box.

Menu Page - example



Menu Page - example

Pyramid

- What

Link together a sequence of pages with Back/Next links. Create a parent page that links to all of the pages in this sequence, and let the user view them either in sequence or out of order.

- Use when

The site or application contains a sequence of items that a user would normally view one after another, such as a slideshow, a wizard, chapters in a book, or a set of products.

Some users would rather view them one at a time and out of order, however, and they need to be able to pick from a full list of the items.

Almost all Picture Managers use a Pyramid navigational model. Sometimes people need to look at pictures individually; sometimes they would rather browse by walking through the whole sequence. Pyramids support both use cases.

Pyramid

- Why

This pattern reduces the number of clicks it takes to get around. It improves navigation efficiency, and it expresses a sequential relationship among the pages.

Back/Next (or Previous/Next) links or buttons are all well and good. People know what to do with them.

But a user doesn't necessarily want to be locked into a page sequence that he can't easily get out of: having gone seven pages in, will he need to click the Back button seven times to get back where he started? Not fun!

By putting a link back to the parent page on each sequence page, you increase the user's options.

You've now got three main navigation options instead of two—Back, Next, and Up.

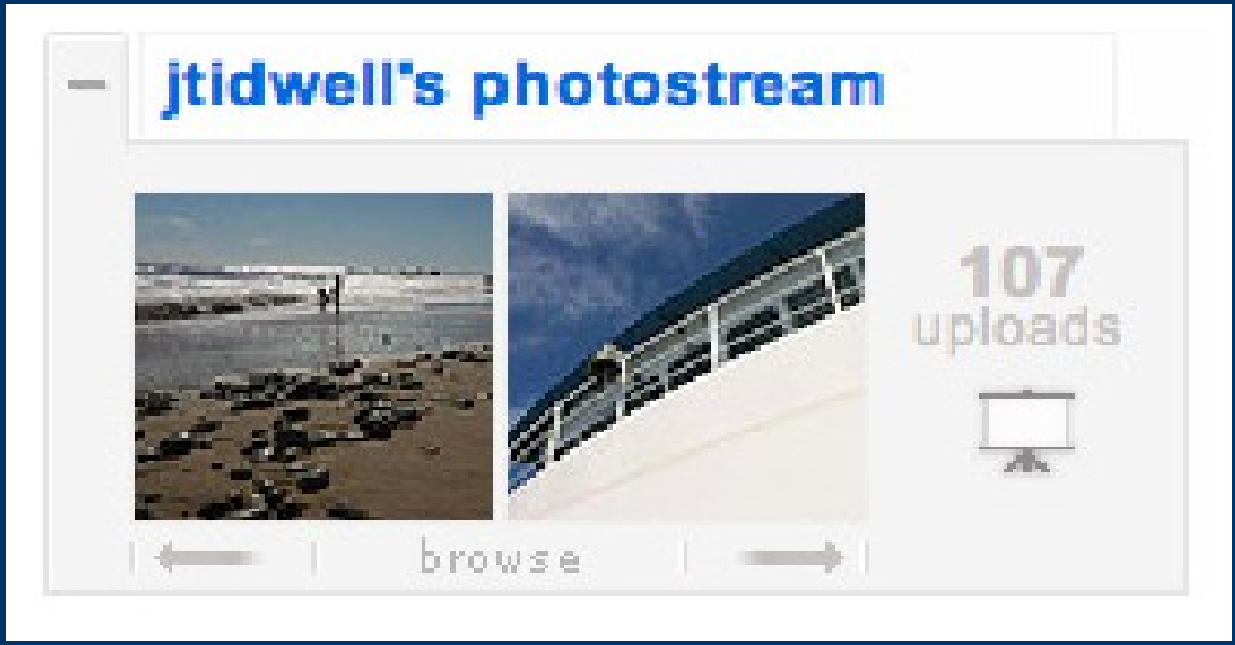
You haven't made it much more complex, but a casually browsing user will need far fewer clicks to go where he wants to go.

Pyramid

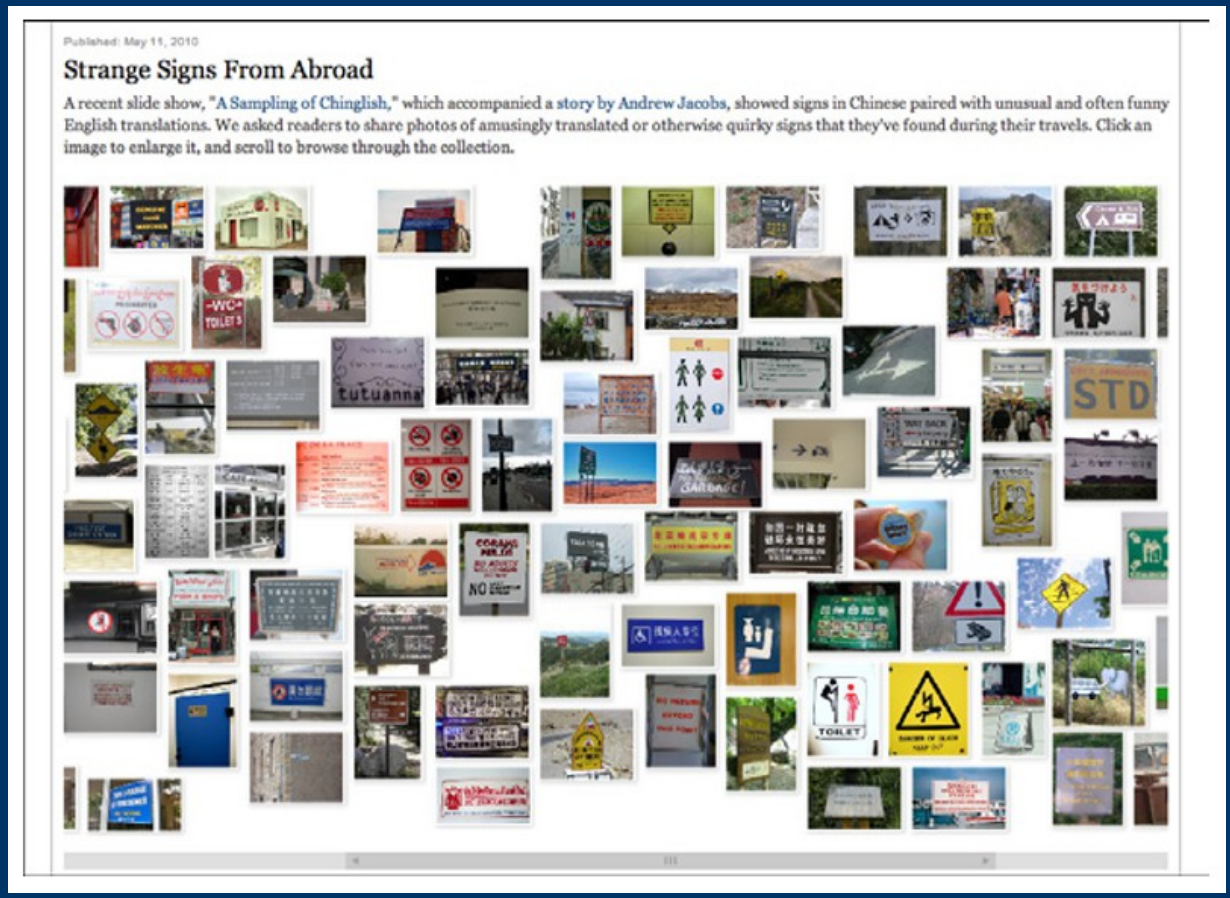
Advice:

- List all the items or pages, in order, on the parent page. Render the list in a way that suits the types of items you're dealing with, such as a Thumbnail Grid for photos, or a rich text list for articles.
- A click on an item or link brings the user to that item's page. On each item page, put Back/Next links. Many sites show a small preview of the next item, such as its title or a thumbnail.
- In addition, put in an Up link to bring the user back to the parent page, and label it with "Back to <Page Title Here>" or something similar.

Pyramid - example



Pyramid - example



Pyramid - example

Published: May 11, 2010

Strange Signs From Abroad

A recent slide show, "A Sampling of Chinglish," which accompanied a story by Andrew Jacc
English translations. We asked readers to share photos of amusingly translated or otherwi
image to enlarge it, and scroll to browse through the collection.

◄ 121 of 176 ► CLOSE ☒

Paris, France

Intersection of the Boulevard du Montparnasse and the Boulevard Saint-Michel.

Taken by Daniel Gallagher

Modal Panel

- What

Show only one page, with no other navigation options, until the user finishes the immediate task.

- Use when

The app or site has gotten into a state from which it shouldn't or can't proceed without input from the user. In a document-centric application, for instance, a "save" action might need the user to supply a filename if one wasn't already given. In other contexts, the user may need to sign in before proceeding, or acknowledge an important message.

If the user simply initiates a minor action that may need further input, try to find a way to ask for that input without a modal panel.

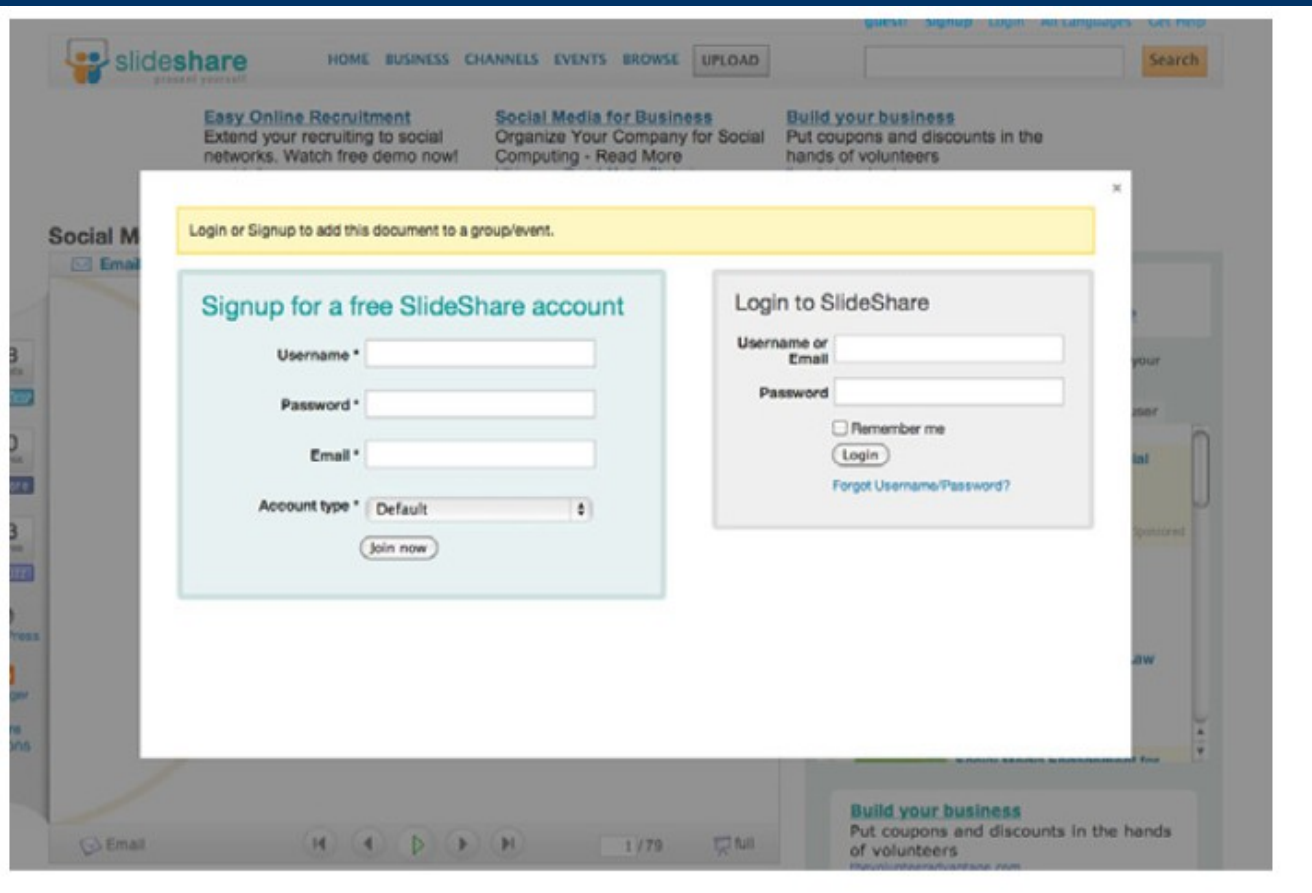
Modal Panel

- Why

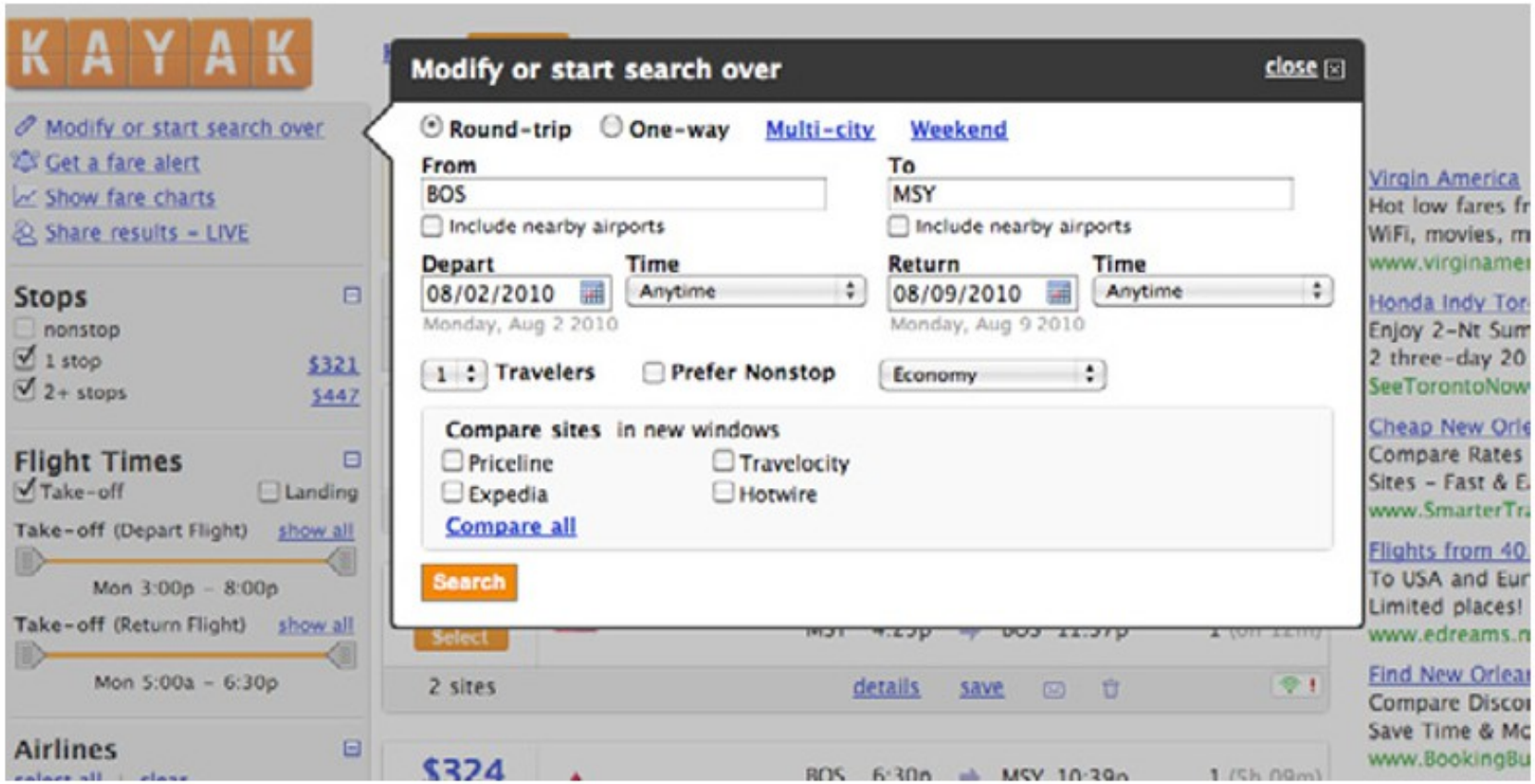
A modal panel cuts off all other navigation options from the user. He can’t ignore it and go somewhere else in the app or site: he must deal with it here and now. When that’s done, he gets sent back to where he was before. It’s an easy model to understand—and to program—though it was overused in applications of past years.

A modal panel is disruptive. If the user isn’t prepared to answer whatever the modal panel asks, it interrupts his workflow, possibly forcing him to make a decision about something he just doesn’t care about. But when used appropriately, a modal panel channels the user’s attention into the next decision that he needs to make. There are no other navigation possibilities to distract him.

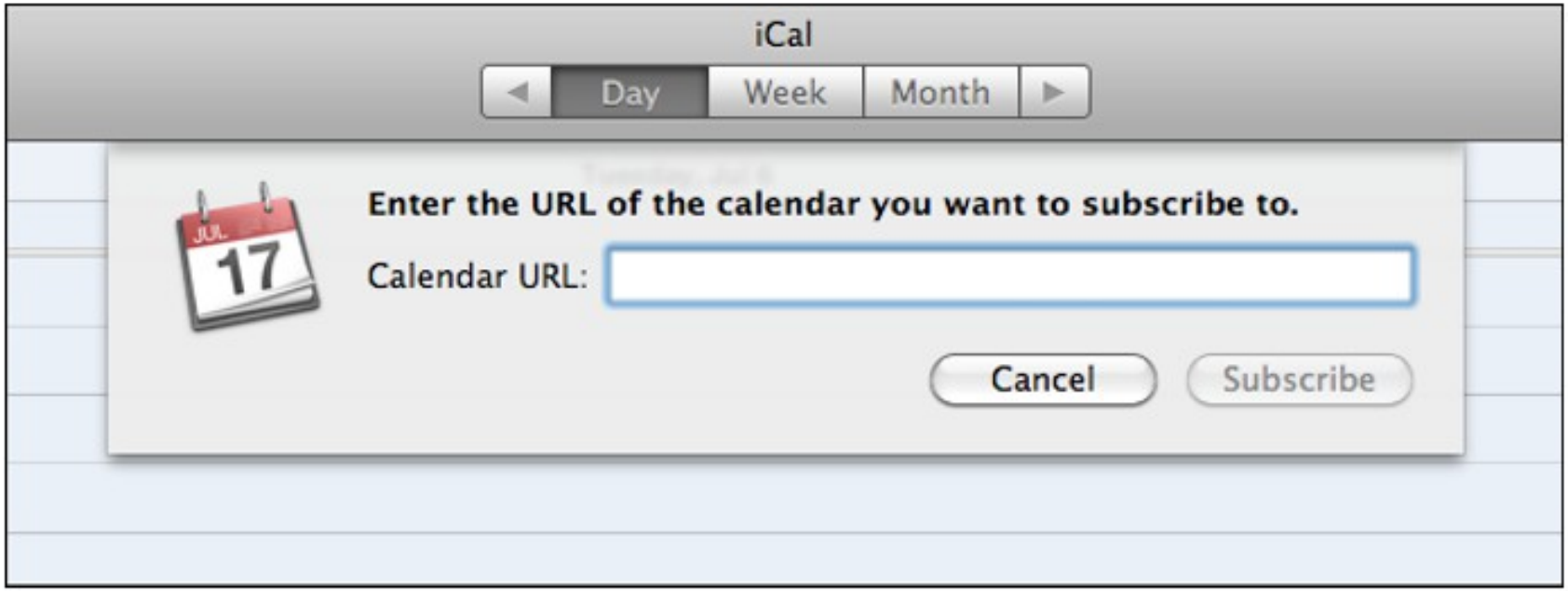
Modal Panel - example



Modal Panel - example



Modal Panel - example



Deep-linked State

- What

Capture the state of a site or app in a URL that can be saved or sent to other people. When loaded, it restores the state of the app to what the user was seeing.

- Use when

The site or app's content is something large and interactive, such as a map, book, video, or information graphic. A specific desired point or state might be hard to find, or it may take many steps to get there from a typical starting point. The app may have many usersettable parameters or states, such as viewing modes, scales, data layers, and so on—these may add to the complexity of finding a particular point and seeing it in the "right" way.

Deep-linked State

- Why

Deep-linked State gives the user a way to jump directly to a desired point and application state, thus saving time and work. It behaves like a “deep link” directly into a piece of content on a conventional site—or a permalink to a blog entry—in the sense that you end up with a URL pointing directly to the desired content.

But it can be more complex than a permalink, because it can capture both application state and content position.

This pattern is useful for saving a state that the user might want to re-create later, especially if he can “bookmark” it using well-known mechanisms (like browser bookmarks, sites such as Delicious, etc.). It’s also handy for sharing with other people, and that’s where it really shines.

A URL representing a Deep-linked State can be emailed, tweeted, posted to a social network, discussed in a forum, published in a blog entry, and talked about in any number of ways. It might make a statement, or go viral, or become a “socially mediated object.”

Deep-linked State

- Advice:

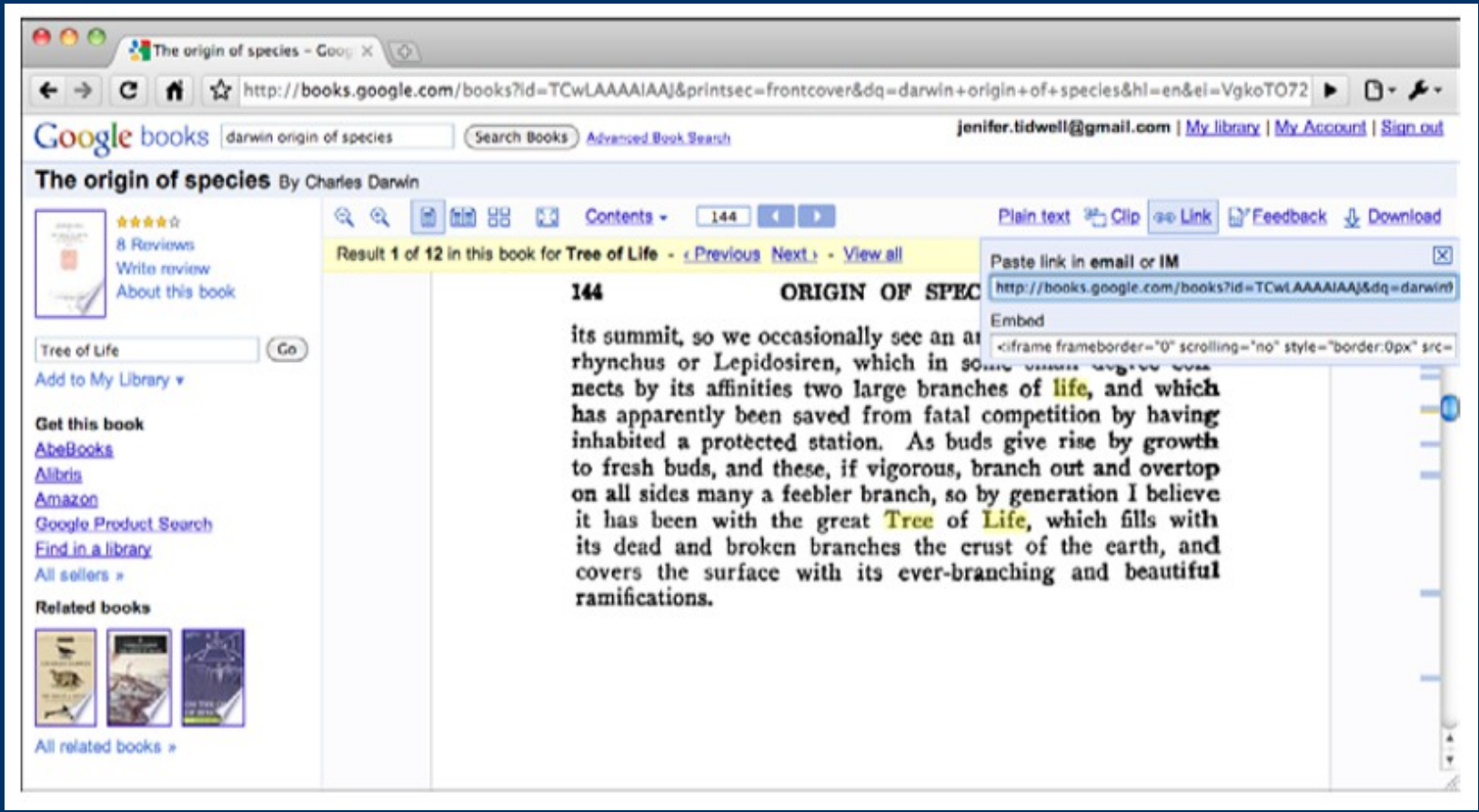
- Track the user's position in the content, and put that into a URL. Track supporting data there as well—comments, data layers, markers, highlighting, and so on—so that reloading the URL will bring it all back.

- Consider what other parameters or interface states you might want users to save: zoom levels, magnification, viewing modes, search results, and so on.

- URLs are the best format for saving Deep-linked States: they are universally understood, portable, short, and supported by a vast variety of tools, such as bookmarking services.

- As a user moves through the content and changes various parameters, immediately put the updated URL in the browser's URL field so that it can be easily seen and captured. Not everyone will think to find it there, so you might also design a "Link" feature whose existence tells the user, "Here's how you create a link to this screen."

Deep-linked State - example



Deep-linked State - example

Deep-linked State - example

Escape Hatch

- What

On each screen that has limited navigation options, place a button or link that clearly gets the user out of that screen and back to a known place.

- Use when

You've got pages that constitute some sort of serial process, such as a wizard, or any pages that lock the user into a limited navigation situation, such as a Modal Panel. These might be pages that users can reach out of context, as they could do via search results.

(Escape Hatches sometimes aren't necessary when you have Sequence Maps or Breadcrumbs on a page. Users who understand them can use those to get back to some known place.).

Escape Hatch

- Why

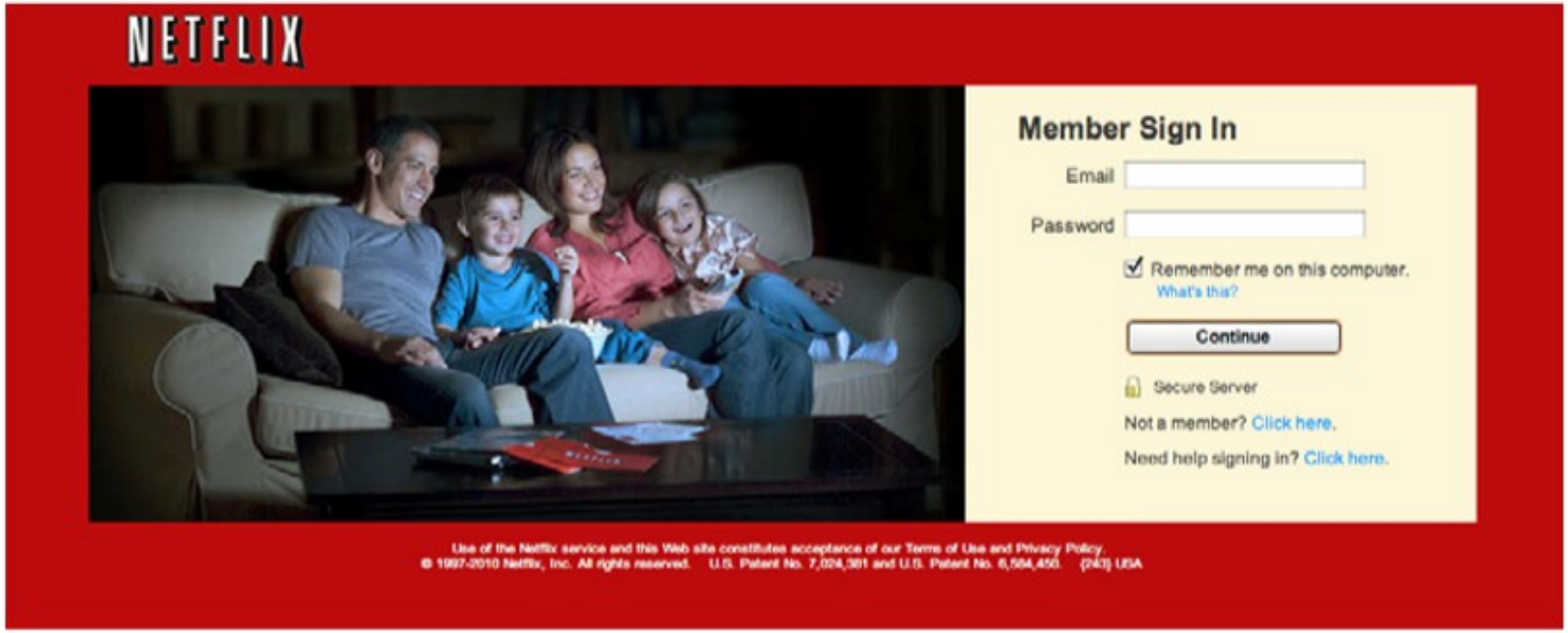
Limited navigation is one thing, but having no way out is quite another! If you give the user a simple, obvious way to escape from a page, no strings attached, he's less likely to feel trapped there.

This is the kind of feature that helps people feel like they can safely explore an app or site. It's sort of like an undo feature—it encourages people to go down paths without feeling like they're committing to them.

Now, if these are pages that users can reach via search results, it's doubly important that Escape Hatches be put on each page. Visitors can click these to get to a "normal" page that tells them more about where they actually are.

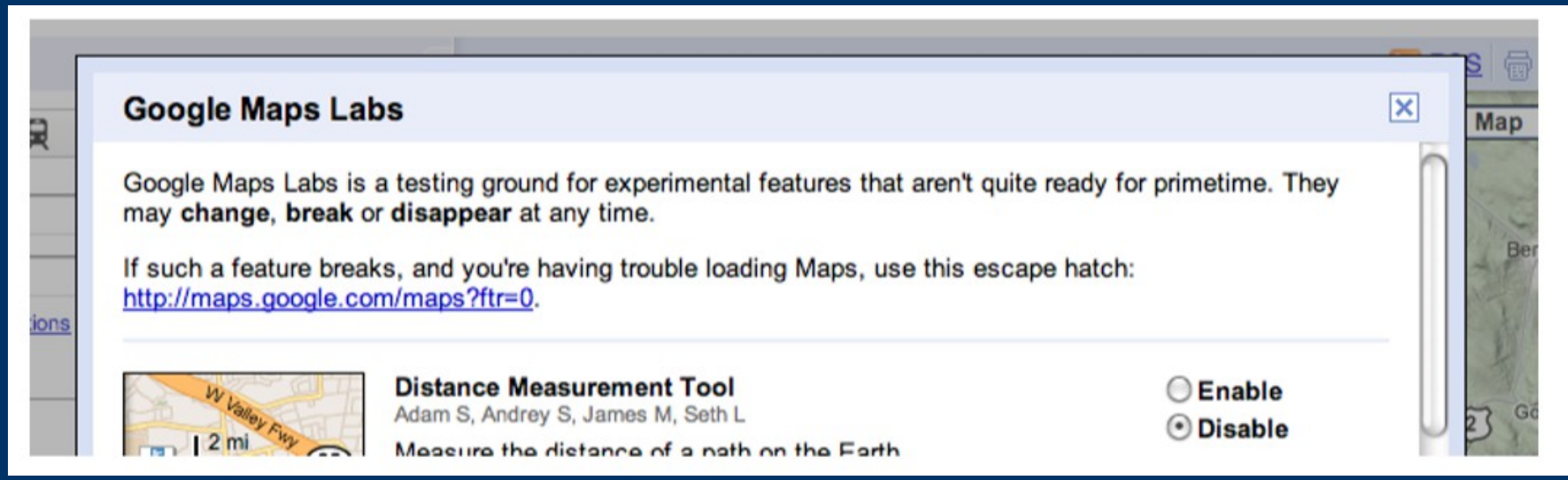
Advice: Put a button or link on the page that brings the user back to a "safe place." This might be a home page, a hub page in a hub-and-spoke design, or any page with full navigation and something self-explanatory on it. Exactly what it links to will depend upon the application's design.

Escape Hatch - example



Netflix sign-in page, with the logo as an Escape Hatch

Escape Hatch - example



Fat Menus

- What

Display a long list of navigation options in drop-down or fly-out menus. Use these to show all the subpages in site sections. Organize them with care, using well-chosen categories or a natural sorting order, and spread them out horizontally.

- Use when

The site or app has many pages in many categories, possibly in a hierarchy with three or more levels. You want to expose most of these pages to people casually exploring the site, so they can see what's available. Your users are comfortable with drop-down menus (click to see them) or fly-outs (roll over them with the pointer).

Fat Menus

- Why

Fat Menus make a complex site more discoverable. They expose many more navigation options to visitors than they might otherwise find.

By showing so many links on every page, you make it possible for a user to jump directly from any subpage to any other subpage (for most subpages, anyhow). You thus turn a multi-level site—where subpages aren't linked to the subpages in other site sections—into a fully connected site.

Fat Menus are a form of progressive disclosure, an important concept in user interface design. Complexity is hidden until the user asks to see it. A visitor to a site that uses these can look over the menu headings to get a high-level idea of what's there, and when he's ready to dive in, he can open up a Fat Menu with a gesture.
If you're already using menus in your global navigation, you might consider expanding them to Fat Menus if surfacing more links makes the content more attractive to casual browsers.

Fat Menus

- Advice:

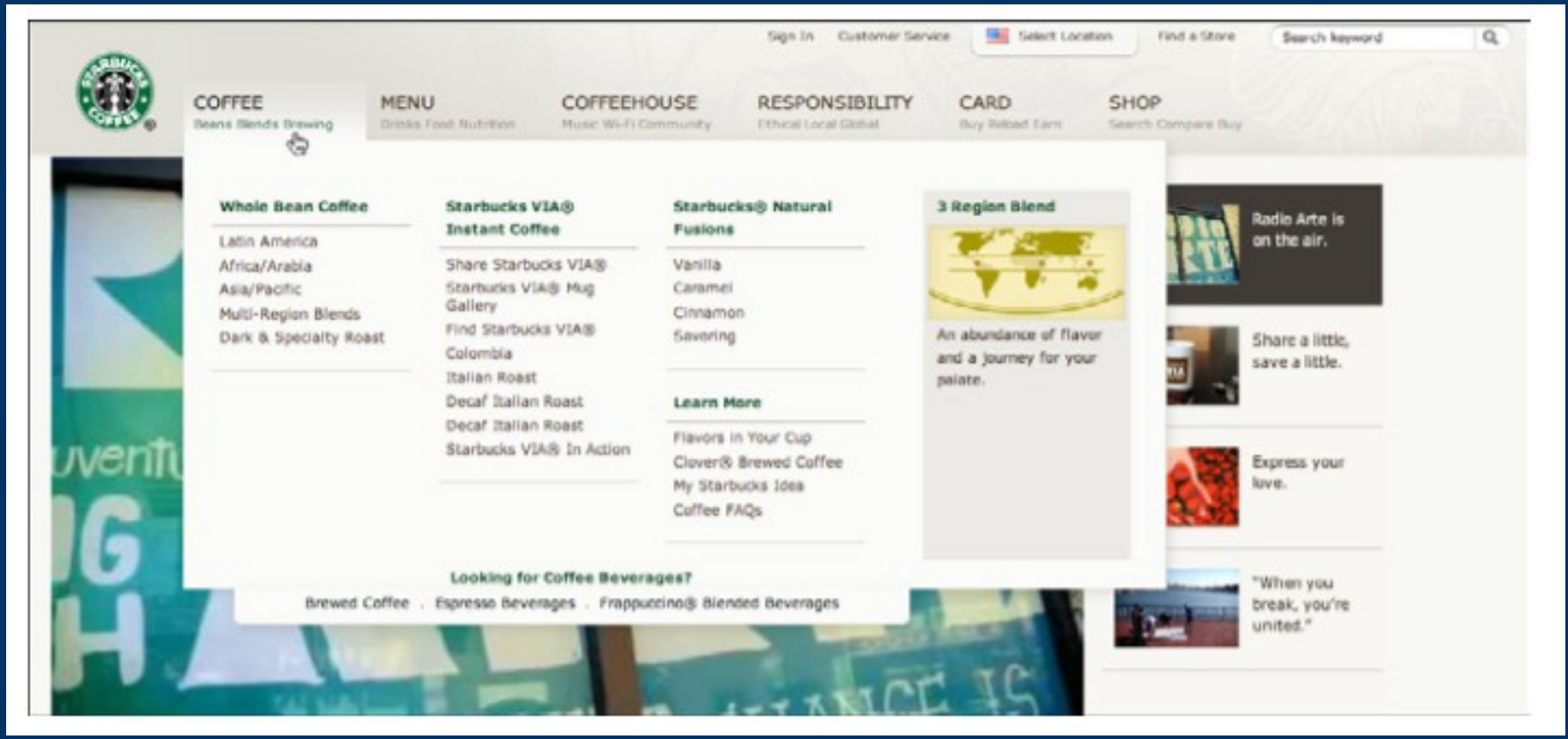
On each menu, present a well-organized list of links. Arrange them into Titled Sections if they fit into subcategories; if not, use a sorting order that suits the nature of the content, such as an alphabetical or time-based list.

Use headers, dividers, generous whitespace, modest graphic elements, and whatever else you need to visually organize those links. And take advantage of horizontal space—you can spread the menu across the entire page if you wish.

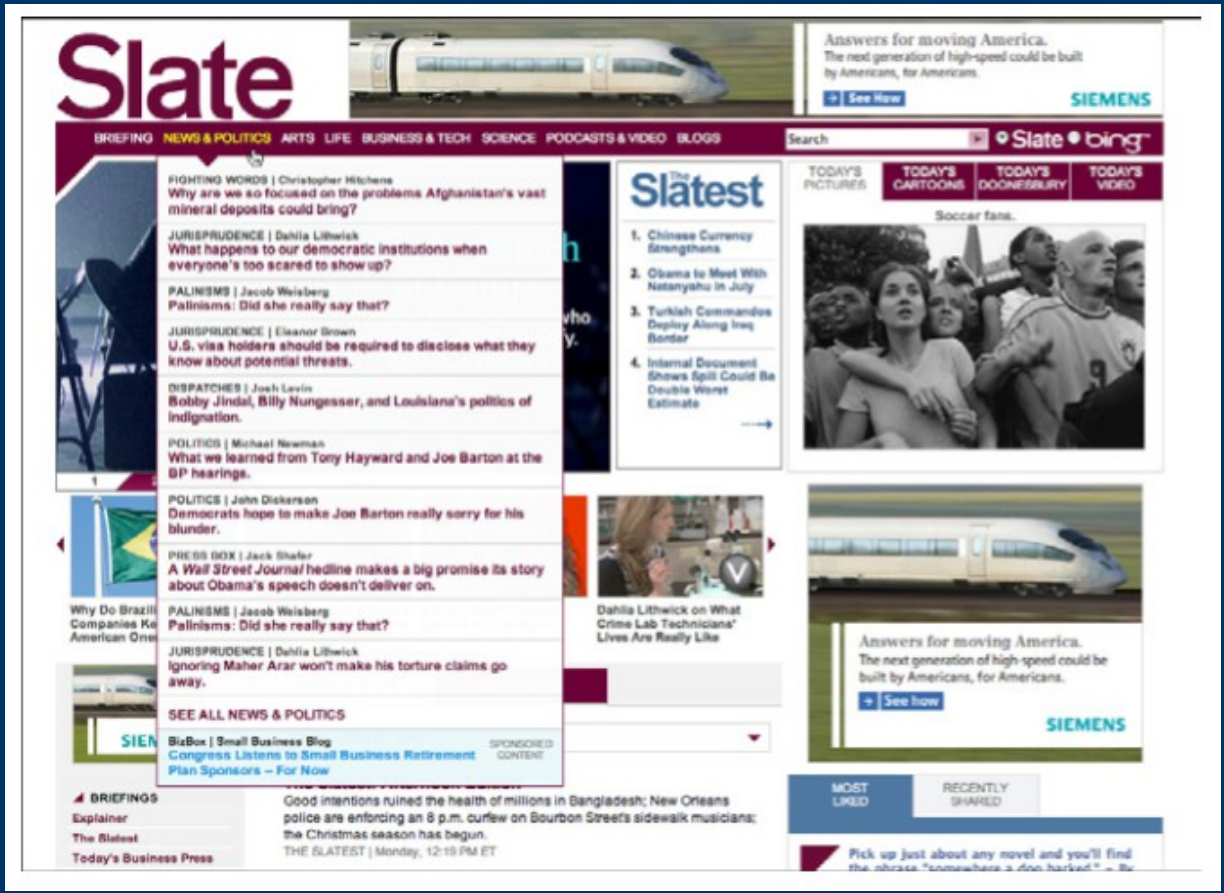
The best sites have Fat Menus that work stylistically with the rest of the site. Design them to fit well into the color scheme, grid, and so on of the page.

Some menu implementations don't work well with accessibility technology such as screen readers. Ensure that your Fat Menus can work with these. If they can't, consider switching to a more static strategy, such as a Sitemap Footer.

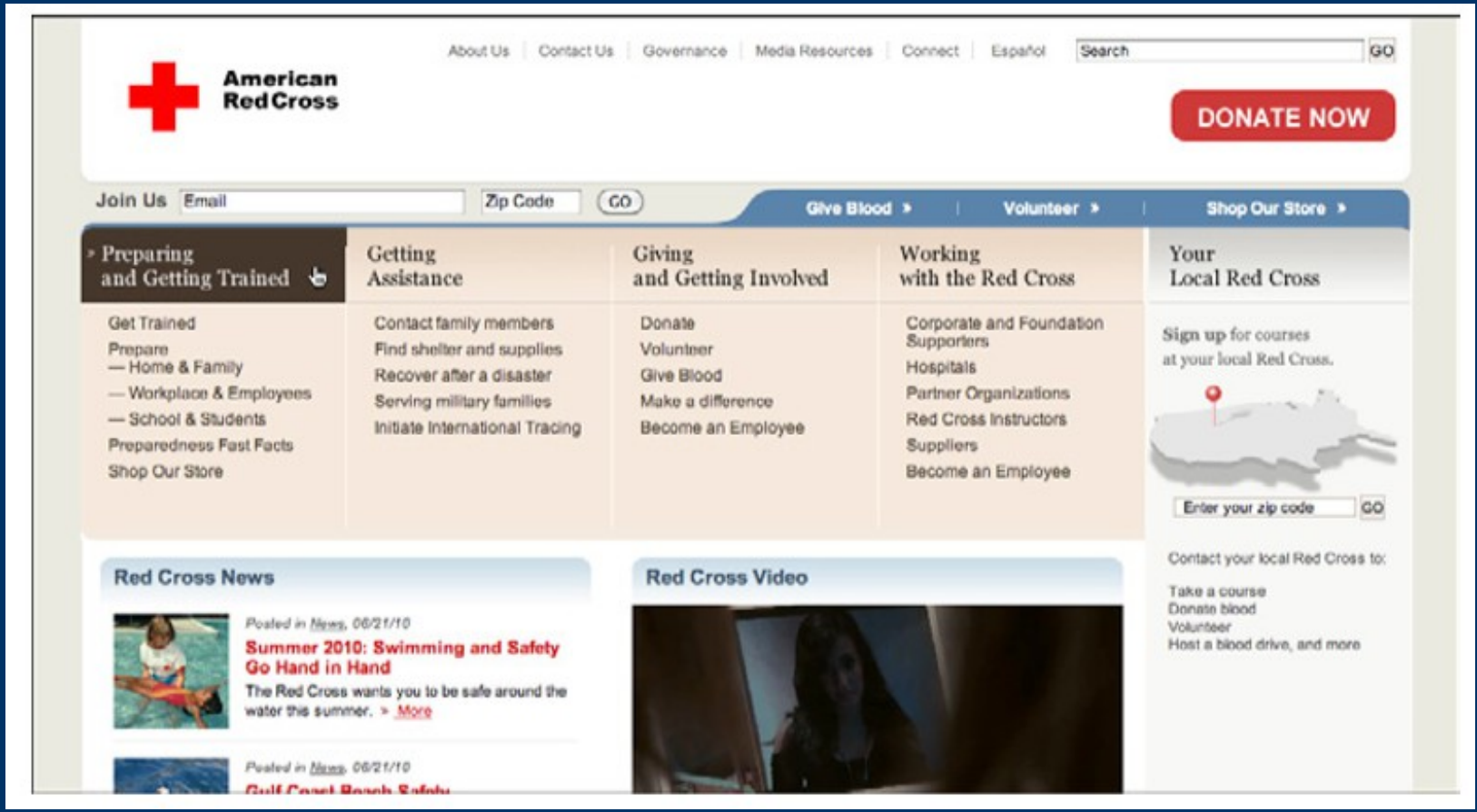
Fat Menus - example



Fat Menus - example



Fat Menus - example



Sitemap Footer

- What

Place a site map into the footer of every page in a site. Treat it as part of the global navigation, complementary to the header. Abridge the site map if you need to make it fit into a compact space.

- Use when

The site you're designing uses a generous amount of space on each page, and you don't have severe constraints on page size or download time. You don't want to take up too much header or sidebar space with navigation.

Sitemap Footer

- Use when

The site has more than a handful of pages, but not an outrageously large number of categories and “important” pages (things that users will look for). You can fit a reasonably complete site map—at least for pages that aren’t in the header—into a strip no taller than about half of a browser window.

There may be a global navigation menu in the page header, but it doesn’t show all levels in the site hierarchy—maybe it only shows the top-level categories. You prefer a simple, well-laid-out footer instead of Fat Menus, perhaps because of implementation ease or accessibility issues.

Sitemap Footer

- Why

Sitemap Footers make a complex site more discoverable. They expose many more navigation options to visitors than they might otherwise have.

By showing so many links on every page, you make it possible for a user to jump directly from any subpage to any other subpage (or major page, anyhow).

You thus turn a multilevel site—where subpages aren't linked to the subpages in other site sections—into a fully connected site.

The footer is where the user's attention lands when she reads to the end of a page. By placing interesting links there, you entice the user to stay on the site and read more.

Sitemap Footer - example

STORES
Find a Store
Shopping Lists
Cooking Classes
Catering
Gift Cards

CAREERS
Search for Jobs
Why Work Here?
Benefits
Careers FAQ

PRODUCTS
Quality Standards
The Whole Deal
Whole Trade
Food Safety
Locally Grown

BLOGS & PODCASTS
Whole Story Blog
CEO John Mackey's blog
Videos
Whole Body Podcasts

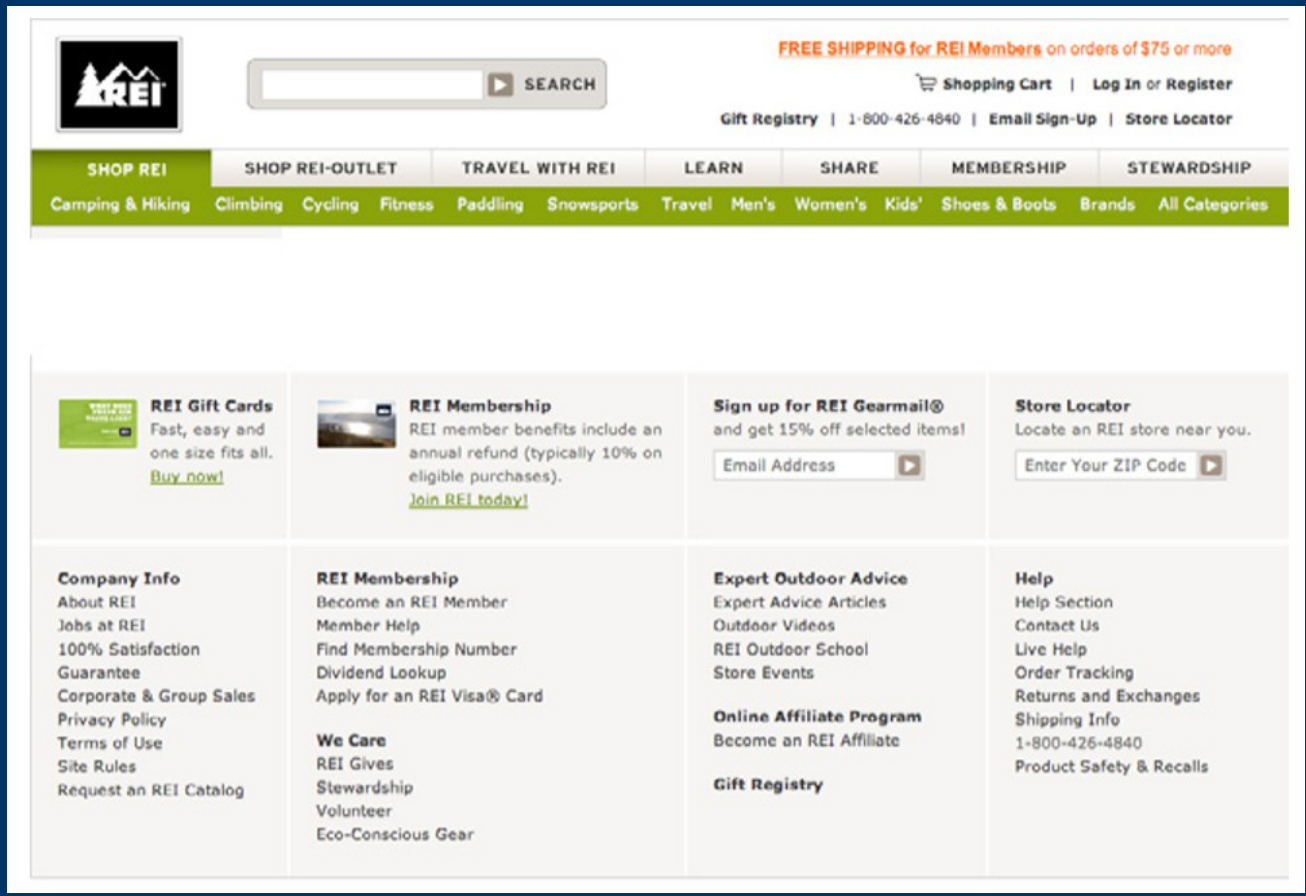
VALUES
Our Core Values
Locally Grown
Green Mission
Whole Planet Foundation
Local Producer Loan Program

COMPANY INFO
History
Board of Directors
Investor Relations
Press Room
Customer Service

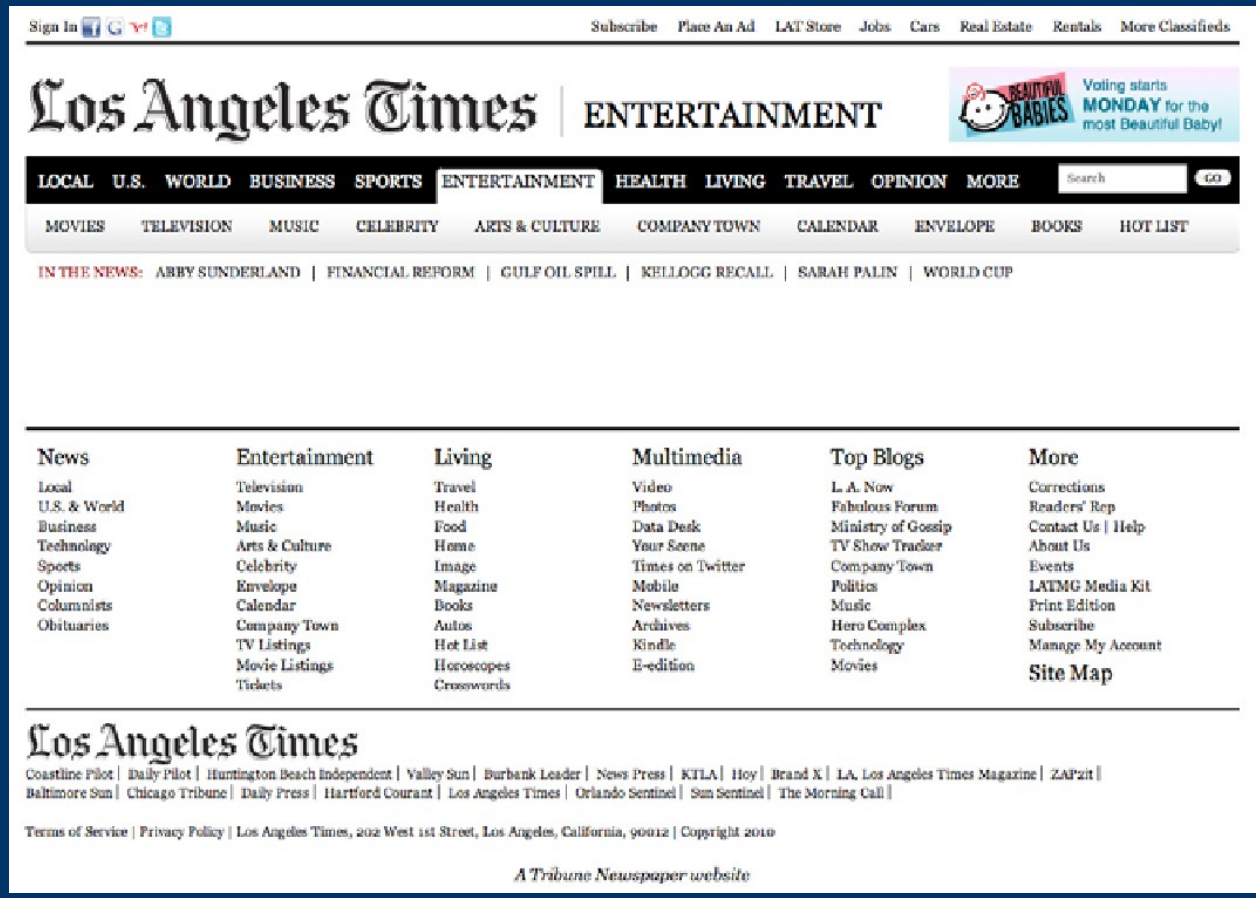
Home | Blog | Site Map | Terms of Use | Privacy Policy | Email Subscriptions | Mobile site

Copyright ©2010. Whole Foods Market IP, L.P.

Sitemap Footer - example



Sitemap Footer - example



Sign In | Subscribe | Place An Ad | LAT Store | Jobs | Cars | Real Estate | Rentals | More Classifieds

Los Angeles Times | ENTERTAINMENT

Voting starts MONDAY for the most Beautiful Baby!

LOCAL | U.S. | WORLD | BUSINESS | SPORTS | ENTERTAINMENT | HEALTH | LIVING | TRAVEL | OPINION | MORE | Search | GO

MOVIES | TELEVISION | MUSIC | CELEBRITY | ARTS & CULTURE | COMPANY TOWN | CALENDAR | ENVELOPE | BOOKS | HOT LIST

IN THE NEWS: ABBY SUNDERLAND | FINANCIAL REFORM | GULF OIL SPILL | KELLOGG RECALL | SARAH PALIN | WORLD CUP

News
Local
U.S. & World
Business
Technology
Sports
Opinion
Columnists
Obituaries

Entertainment
Television
Movies
Music
Arts & Culture
Celebrity
Envelope
Calendar
Company Town
TV Listings
Movie Listings
Tickets

Living
Travel
Health
Food
Home
Image
Magazine
Books
Autos
Hot List
Horoscopes
Crosswords

Multimedia
Video
Photos
Data Desk
Your Scene
Times on Twitter
Mobile
Newsletters
Archives
Kindle
E-edition

Top Blogs
L.A. Now
Fabulous Forum
Ministry of Gossip
TV Show Tracker
Company Town
Politics
Music
Hero Complex
Technology
Movies

More
Corrections
Readers' Rep
Contact Us | Help
About Us
Events
LATMG Media Kit
Print Edition
Subscribe
Manage My Account

Site Map

Los Angeles Times

Coastline Pilot | Daily Pilot | Huntington Beach Independent | Valley Sun | Burbank Leader | News Press | KTLA | Hoy | Brand X | LA. Los Angeles Times Magazine | ZAP2it | Baltimore Sun | Chicago Tribune | Daily Press | Hartford Courant | Los Angeles Times | Orlando Sentinel | Sun Sentinel | The Morning Call |

Terms of Service | Privacy Policy | Los Angeles Times, 202 West 1st Street, Los Angeles, California, 90012 | Copyright 2010

A Tribune Newspaper website

Sign-in Tools

- What

Place utility navigation related to a signed-in user's site experience in the upper-right corner. Show tools such as shopping carts, profile and account settings, help, and sign-out buttons.

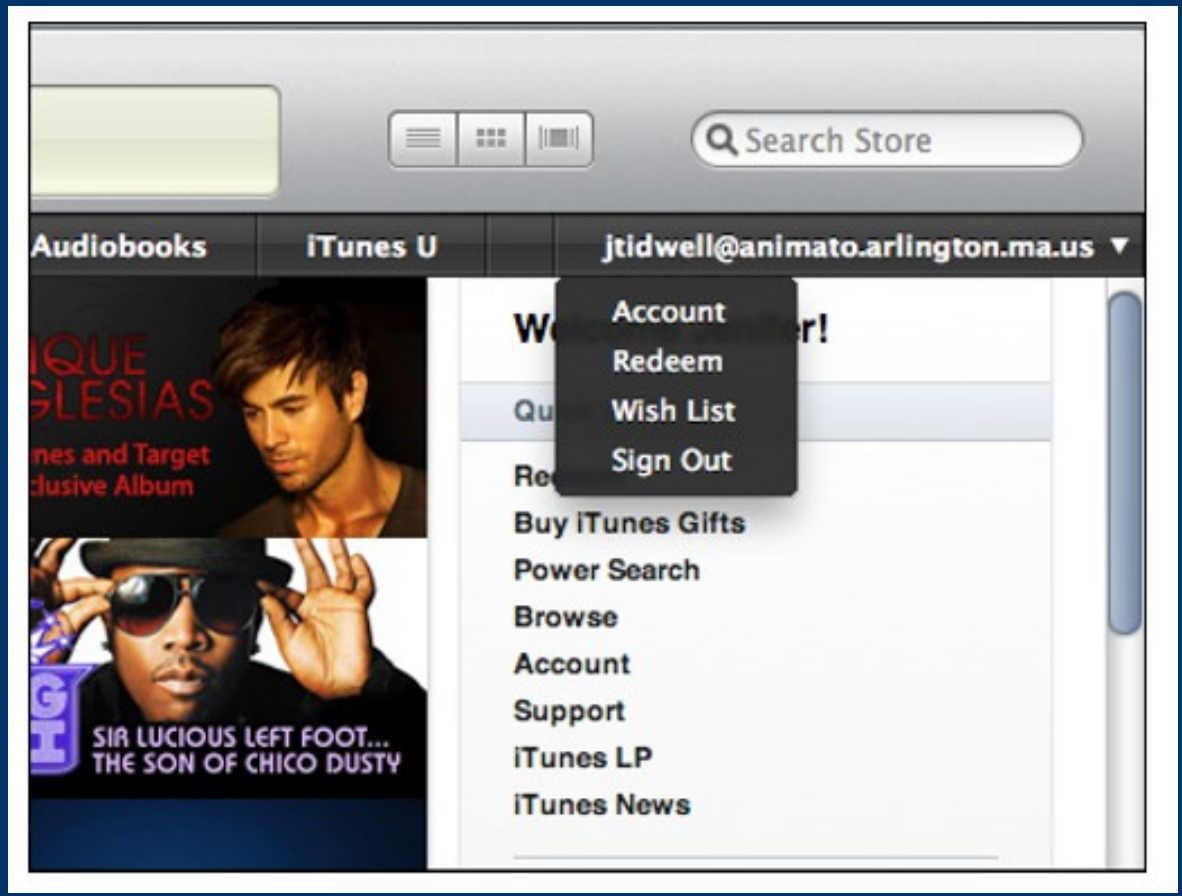
- Use when

Sign-in Tools are useful for any site or service where users often sign in.

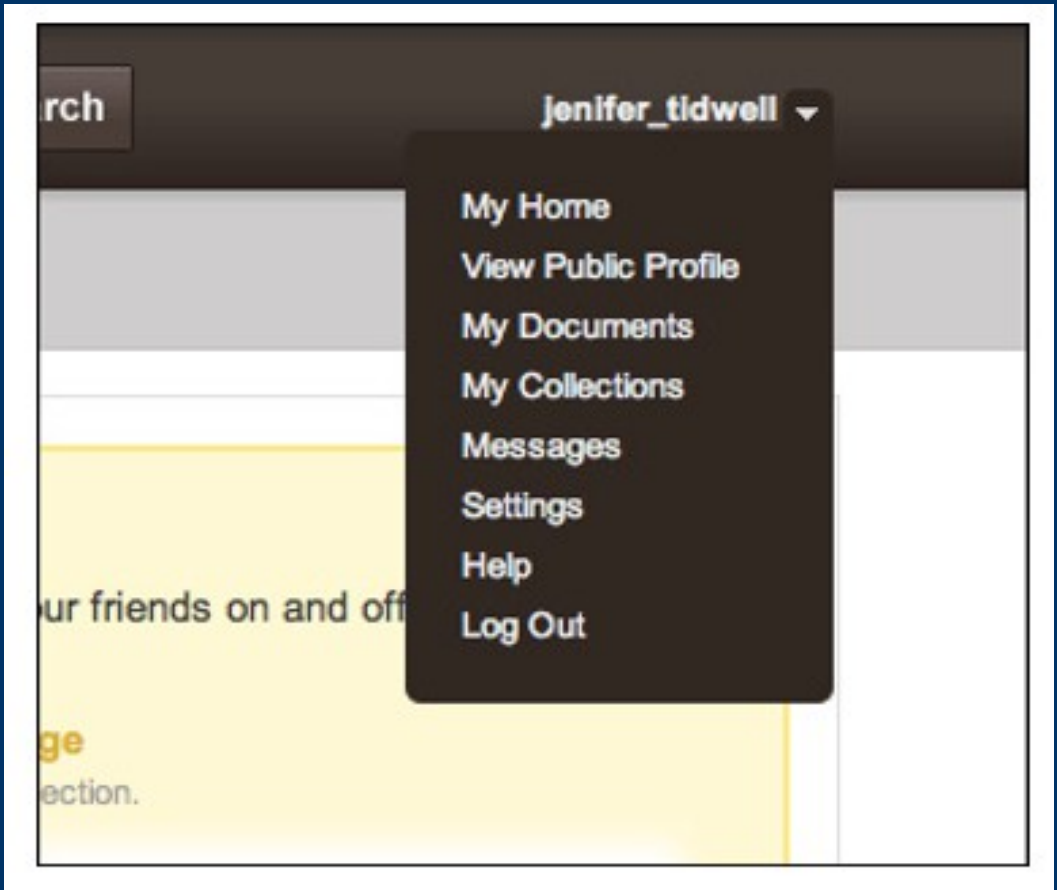
- Why

This pattern is purely convention; the upper-right corner is where many people expect such tools to be, so they will often look there. Give users a successful experience by putting these tools where they expect them to be.

Sign-in Tools - example



Sign-in Tools - example



Sequence Map

- What

On each page in a sequence, show a map of all the pages in order, including a "You are here" indicator.

- Use when

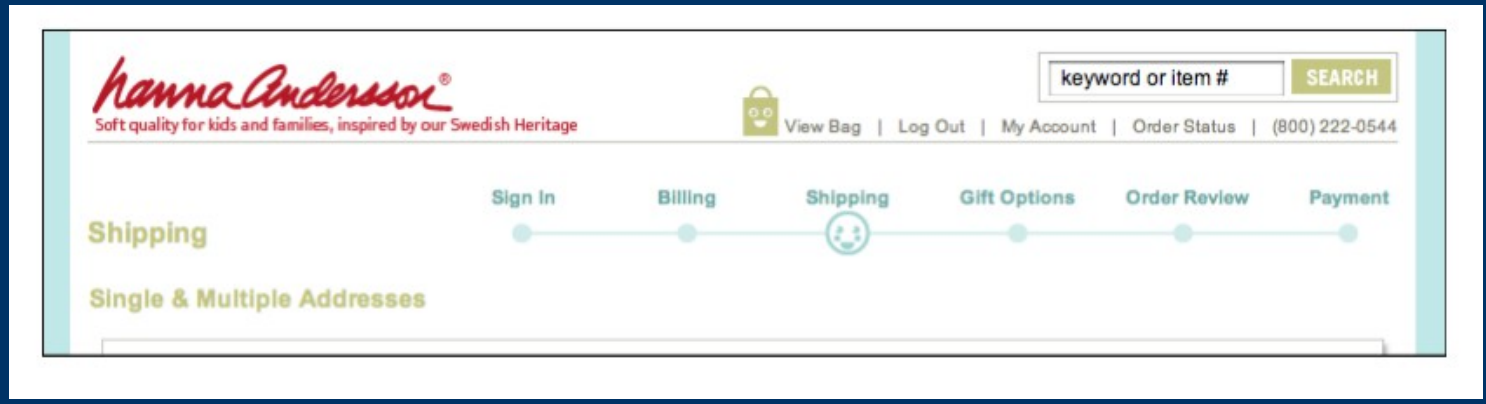
You design a written narrative, a process flow, a Wizard, or anything else through which a user progresses page by page. The user's path is mainly linear.
If the navigation topology is large and hierarchical (as opposed to linear) you may want to consider using Breadcrumbs instead. If you have a large number of steps or items and their order doesn't matter much, this morphs into a Two-Panel Selector or Overview Plus Detail.

Sequence Map

- Why

Sequence Maps tell a user how far he's come through a series of steps—and, more importantly, how far he has yet to go before he's finished. Knowing this helps him decide whether to continue, estimate how long it will take, and stay oriented.

Sequence Maps also serve as navigational devices. If someone wants to go back to a previously completed step, he can do so by clicking that step in the map.



Sequence Map - example

Breadcrumbs

- What

On each page in a deep navigational hierarchy, show a list of all the parent pages, up to the main or home page.

- Use when

Your application or site has a hierarchical structure with two or more levels. Users move around via direct navigation, browsing, filtering, searching within the site, or deep-linking into it from elsewhere.
Global navigation alone isn't sufficient to show a "You are here" signpost, because the hierarchy is too deep or large.
Alternatively, your site or app may have a set of browsing and filtering tools for a large data set, such as products being sold online.
The products are categorized in a hierarchy, but that categorization doesn't necessarily match the way people will look for those products.

Breadcrumbs

- Why

Breadcrumbs show each level of hierarchy leading to the current page, from the top of the application all the way down. In a sense, they show a single linear "slice" of the overall map of the site or app.

So, like a Sequence Map, Breadcrumbs help a user figure out where he is.

Unlike a Sequence Map, though, Breadcrumbs don't tell the user where he's headed next. They deal only with the present.

Instead, Breadcrumbs are best for telling you where you are relative to the rest of the app or site—it's about context, not just history.

Finally, Breadcrumbs are usually clickable links or buttons. This turns them into a navigational device in their own right.

Breadcrumbs

- Advice:

Near the top of the page, put a line of text or icons indicating the current level of hierarchy.

Start with the top level; to its right, put the next level and so on down to the current page. Between the levels, put a graphic or text character to indicate the parent/child relationship between them. This is usually a right-pointing arrow, triangle, greater-than sign (>), slash (/), or right angle quotes (»).

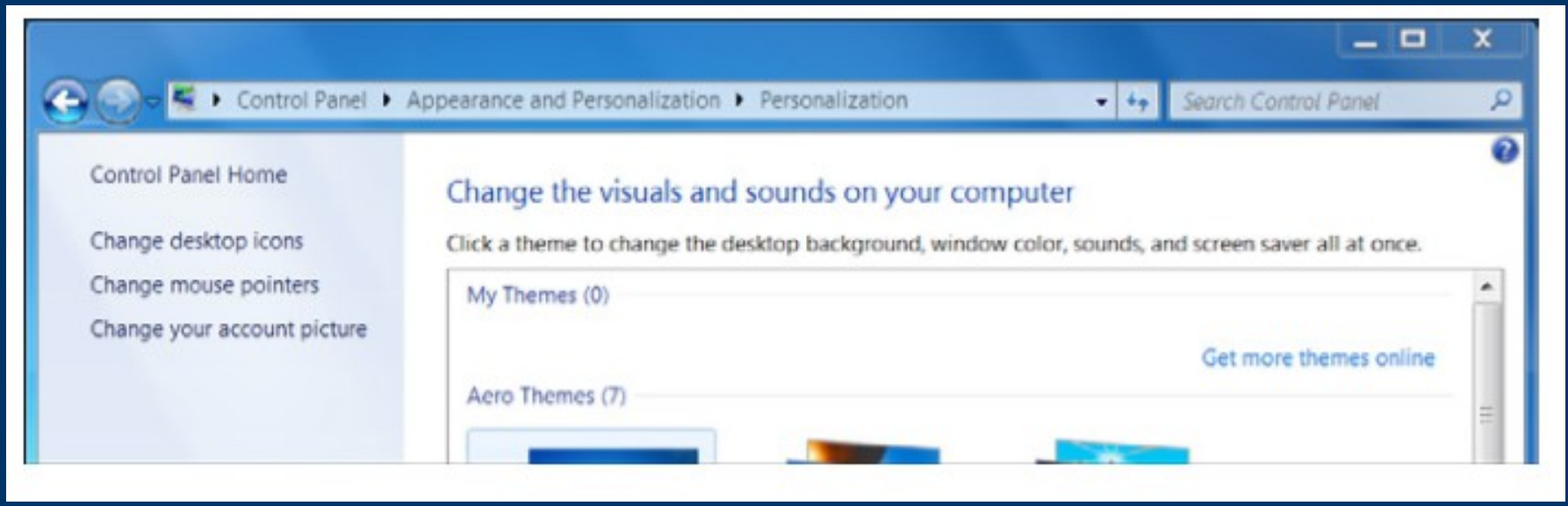
The labels for each page should be the page titles.

The labels should be links to those pages.

Some Breadcrumbs show the current page as the last item in the chain; some don't. If yours do, make them visually different from the rest of the items, since they're not links.

Breadcrumbs - example

Breadcrumbs - example



Breadcrumbs - example

Annotated Scrollbar

- What

Make the scrollbar serve double-duty as a map of the content, or as a “You are here” indicator.

- Use when

You’re designing either a document-centric application or a pan-and-zoom interface, such as a map or large visualization. Users will scan this document or graphic for items of note, such as specific page numbers or landmarks. They might have trouble keeping track of where they are and where to go next as they scroll.

Annotated Scrollbar

- Why

Even though the user remains within one navigational space as she scrolls through the content, signposts are still useful.

When scrolling quickly, it's really hard to read the text flying by (or impossible, if the screen can't refresh quickly enough), so some other indicator of position is necessary.

Even if she stops briefly, the part of the document she can see may not contain anything she can orient herself by, like headers.

Why a scrollbar? Because that's where the user's attention is focused. If you put signposts there, the user will see them and use them as she scrolls, rather than trying to look at two different screen areas at once.

You can put signposts close to the scrollbar and still get the same effect; the closer, the better.

Annotated Scrollbar

- Advice:

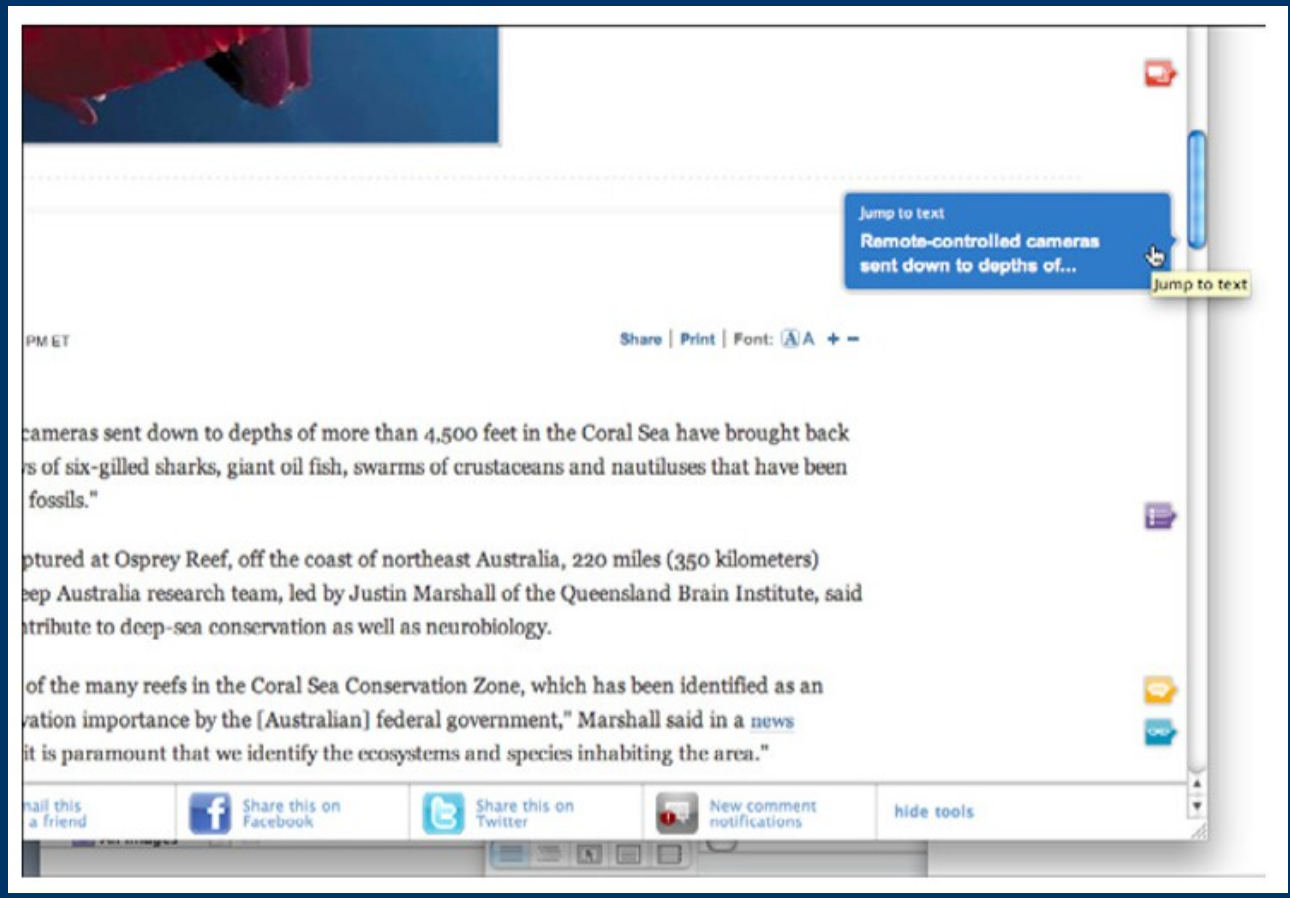
Put a position indicator on or near the scrollbar. Either static or dynamic indicators might work.

Static indicators are those that don't change from second to second, such as blocks of color in the scrollbar track. Make sure their purpose is clear, though; such things can baffle users that aren't used to seeing graphics in the scrollbar track!

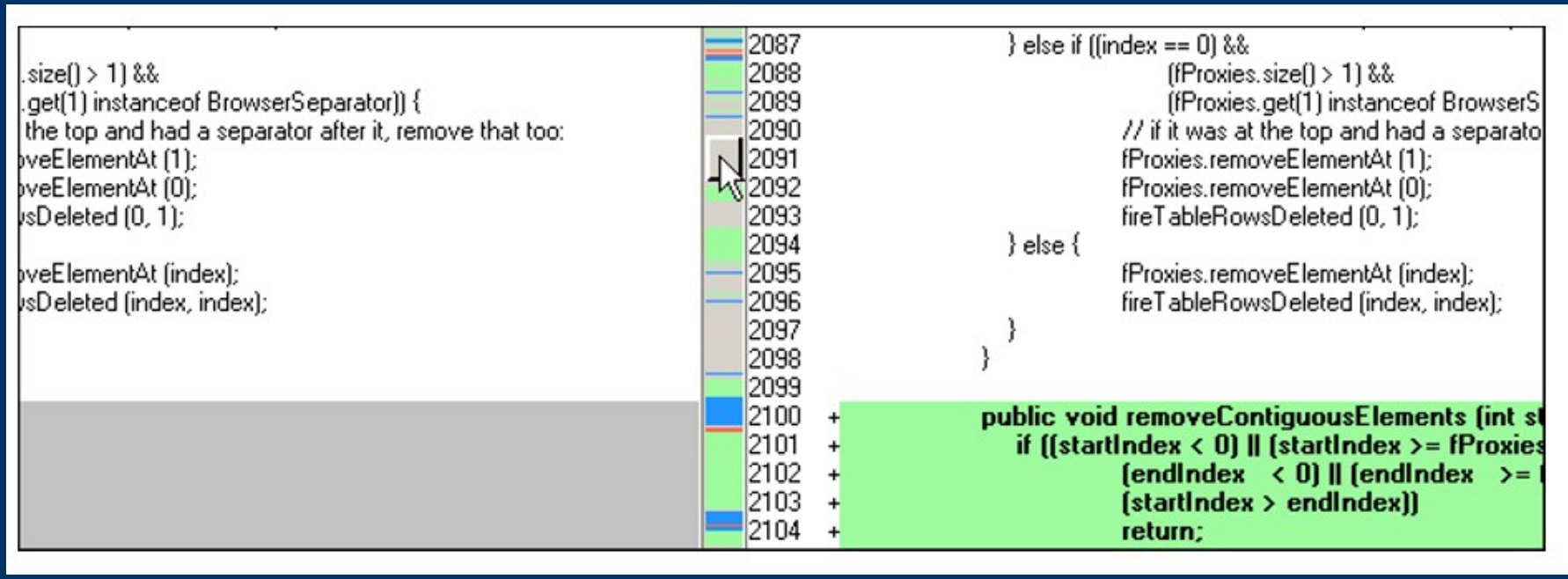
Dynamic indicators change as the user scrolls, and they are often implemented as tool tips. As the scroll position changes, the tool tip shown next to the scroll thumb changes to show information about the content there. This will vary with the nature of the application.

In either case, you'll need to figure out what a user will most likely be looking for, and thus what you need to put into the annotations.

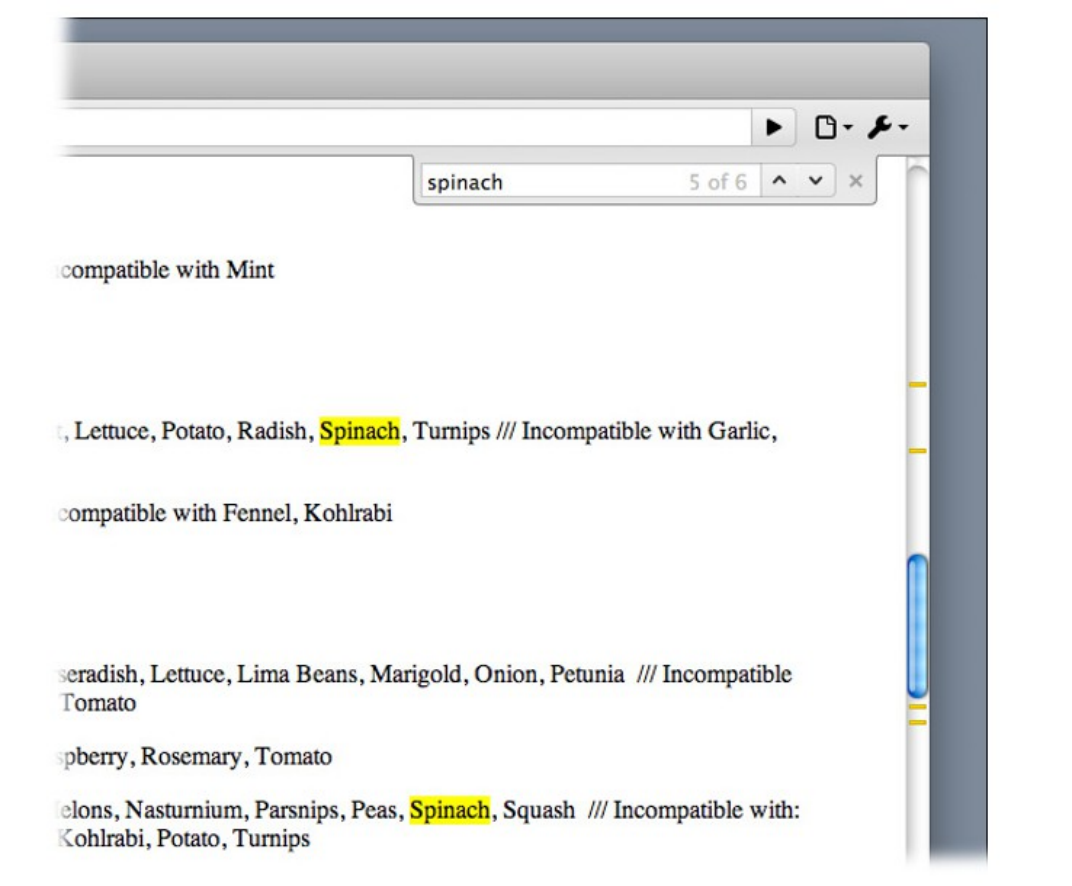
Annotated Scrollbar - example



Annotated Scrollbar - example



Annotated Scrollbar - example



Animated Transition

- What

Smooth out a startling or dislocating transition with an animation that makes it feel natural.

- Use when

Users move through a large virtual space, such as an image, spreadsheet, graph, or text document. They might be able to zoom in to varying degrees, pan or scroll, or rotate the whole thing. This is especially useful for information graphics, such as maps and plots.

Alternatively, the interface might have sections that can be closed and opened again, either by the system or by the user—such as trees with closable parent nodes, standalone windows that open and close, or an interface built with Collapsible Panels.

Animated Transition might also be used when users jump from one separate page to another.

Animated Transition

- Advice:

Some of the types of transitions listed by the Yahoo! pattern library (http://developer.yahoo. com/ypatterns/richinteraction/transition/) and Designing Web Interfaces are as follows:

• Brighten and dim;

• Expand and collapse;

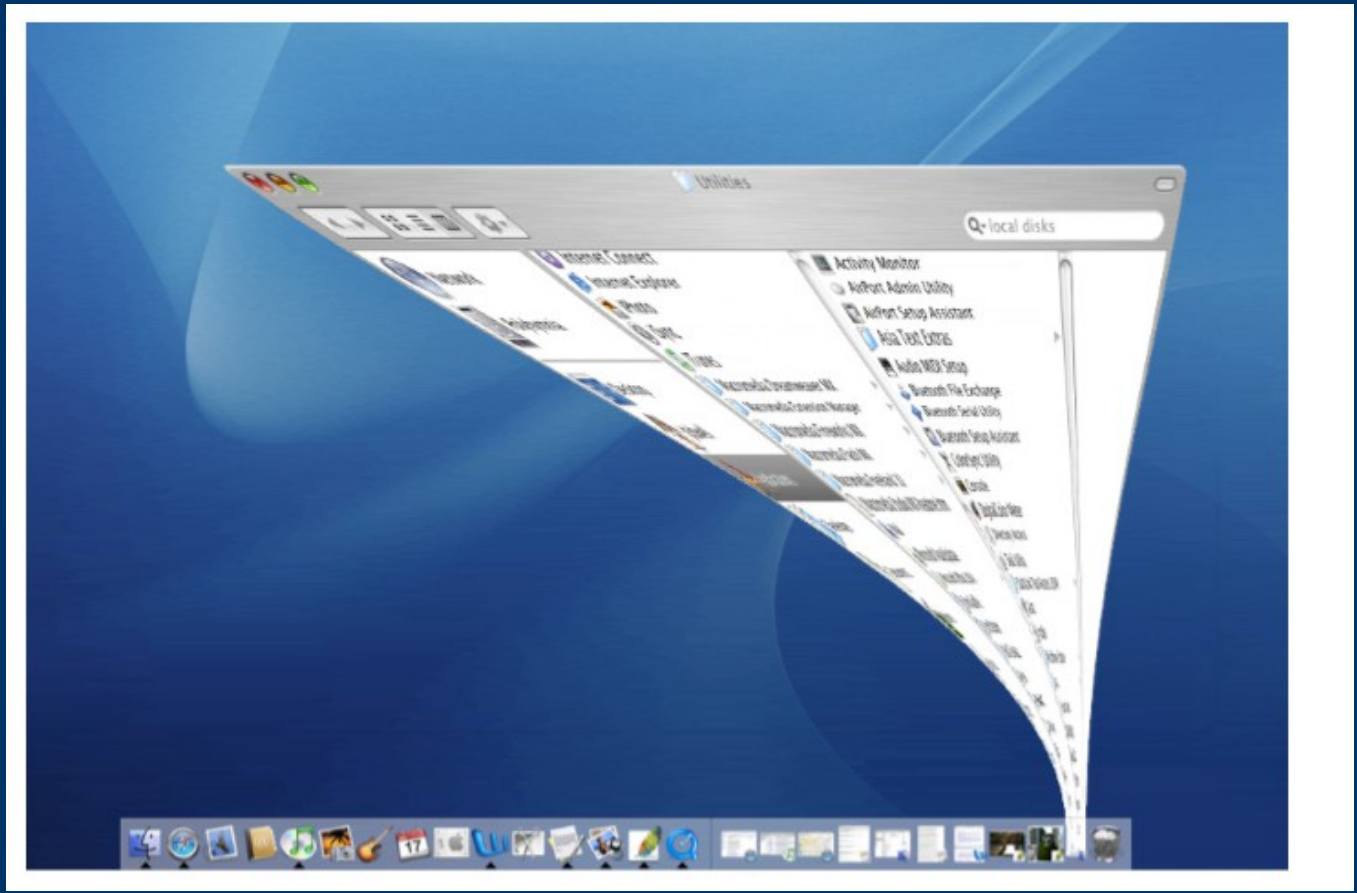
• Fade in, fade out, and cross-fade;

• Self-healing;

• Slide;

• Spotlight.

Animated Transition - example



Thank You for yours attention.