The Habari Administration Interface

Human Interface Guidelines

The Habari Design Manifesto

Inside out, Habari is being designed to be modern, elegant and user friendly. The administrative interface is representative of this approach, and is meant to provide you with the path of least resistance to your goal, while still letting you retain fine-grained control.

Most of what is described in this document, by its very nature should be intuitive and self-explanatory. However, in an effort to maintain these hard-won properties at the benefit of all involved--users and developers alike--this document serves as a guide for working on or adding to Habari's administration interface.

Habari's compartmentalized approach to development dictates that the core of Habari contain only the most essential functionality, leaving bells and whistles to be provided through its powerful plugin architecture.

This mindset should be kept at the forefront of the interface design aspects of Habari as well.

For developers contributing to the core of Habari's administration interface, this means staying the path, and not giving in to the urge to force upon it, features it doesn't need.

For third-party developers, this means focusing on doing a few things great, rather than a lot of things averagely.

For everyone involved, it means honoring the spirit of Habari's powerful simplicity.

General Guidelines

There are some basic rules that will help integrate most plugin option pages, new media silos and the like into the existing style of the interface:

1. Retain Existing Practices
2. Reuse Elements
3. Less is More

Layout structurally, most elements, be it text or form elements, are held inside a centered column of containers. While this provides structure and segregation, it is generally a good rule of thumb for the elements themselves to imply the structure.

A Word on the Use of Colors

The majority of Habari's administration interface is kept in the neutral shades of grayscale, with colors being used only when truly needed, or at the very least when they distract the least. This is done to keep the administration interface 'genderless', and to promote the use of shapes and contrast as the primary aides in guiding the user.

Therefore, please consider with great care, the addition of color to an element. Does the color serve a user-centric purpose? Does it enhance the use of the elements, or would it do just as well being grayscale? Does it call unwanted attention to itself, at the cost of other, perhaps more important elements?

Related to the use of colors is the use of flat, graded and pseudo-3D elements. Generally speaking, gradients aren't used for anything except to indicate shadows, leaving the vast majority of the elements in the Habari administration interface flat-shaded.

A Word on Blueprint CSS

The Habari administration interface makes use of a CSS framework called Blueprint CSS to encourage a more coherent and homogeneous look and feel. You can read more about Blueprint CSS and its many virtues elsewhere. Suffice it to say, that it is good practice to stay as much as possible within the confines of its column system whenever possible.

If you do need to lay out differently from what Blueprint CSS was made for, consider if there exists a precedence for what you're trying to achieve, and consider reusing that solution instead, if possible.

A Word on Width

There are basically three widths in Habari's administration interface:

- Single column, 790px wide.
  This is how most of Habari's administration interface is laid out, including the create, options and manage pages. Most of the content should be kept in individual containers, with new containers for each set of content (as per the Configure pages).
- Multiple column, xxxpx wide.
  For pages like 'Activity', where the purpose of the page is to present the user with an overview of the site's activity, the data ´might be best presented in multiple columns, both to increase the data intake, but also to save the user from having to scroll, thus speeding up the time it takes to gain an overview.
- Full-width pagesplitter.
  While using it for full-width elements isn't mandatory, the pagesplitter was designed specifically for this particular use, like for instance the timeline on the manage pages or the media silo plugins. A more in-depth description of its use and operation can be found later in this document.

As with everything else, it is suggested that you follow examples already set by the interface layout, if at all possible. And if that isn't possible, to at least stay in the 'spirit' of the layout.

The Menu Bar

Habari's near-black menu bar stays fixed to the top of the browser window, regardless of scrolling, making it always available to the user.. It contains to its extreme left, the menu itself, which is described below. And to its extreme right, a link to and the name of, the site for which this is the administration interface.

The Menu

The function of The Menu isn't that different from that of the Windows Start button; namely to provide a single point of entry to the entirety of the Habari administration interface. As such any and all pages available in the administration interface are, and should be, available from this menu.

The Menu is a 'dropbutton', which serves both as a menu as well as a title for the page you are currently on. Dropbuttons are described in greater detail later in this document, but the basic gist is that upon hovering the dropbutton, a menu will reveal itself.

One thing differentiates The Menu from the rest of the dropbuttons in Habari; it has sub-menus. Also worth noting is that the parents of these sub-menus are themselves proxy links of the first item in the sub-menu. Because of this behavior, items in a sub-menu should obviously be directly related to the sub-menu's parent, with the most related and most likely to be used item, at the very top.

Aside from appearing as the label on the dropbutton itself, function thus as a title for the current page, the item for the current page is also marked in The Menu listing itself by a small 'V' mark, to differentiate it.

The 'Content/Admin' Menu Structure

All the available pages in the Habari administration interface fall into one of three categories:

- **Content**: Create, Comments, Entries, Pages and Tags
  Habari's raison d'être is the creation, publication and management of content. Whether it be entries, comments or tags.
- **Admin**: Activity, Configure, Themes, Plugins, Import, Users and Logs
  Anything not directly related to the creation and management of content is Admin, including the configuration of Habari itself, activating and deactivating themes or plugins, creating and administrating users.

At the bottom of The Menu, the item 'Logout' is to be found. It falls outside of the category system.

Adding Custom Pages to the Menu

Not only does the addition of new pages to The Menu confuse the users, but The Menu itself can only hold so many items before it becomes confusing, and thus inefficient. Adding new pages to the administration page structure is therefore generally to be avoided unless absolutely necessary.

Habari has a framework in place which allows plugins and themes what essentially amounts to their own page, nestled in the confines of the Configure 'section' of the administration structure.

It is understood however, that some plugins might service the user better by being more readily available, for instance by having a place in the main menu itself. If this is the case, please consider with great care where the menu item best serves the user, rather than where it might get the most exposure.

Use the above section on the Content/Admin structure to determine where your page would work best, and whether or not it might best serve as in a sub-menu.

Custom Controls

To help make Habari's administration interface as efficient as possible, a few custom controls have been created, the operation of which should be self-evident, but the use of which might require some guidance.

The Dropbutton

To conserve space and speed up the decision-to-action response time, a new control was devised, which operates as a cross between a dropdown menu and a button. Here's how it works:

- When inactive, the dropbutton has a the topmost item in its list as its label.
- Hovering the dropbutton will immediately reveal its list and allow it to be navigated.
- Clicking the dropbutton itself, will activate the topmost link.
- Clicking any link in the dropdown, will obviously activate said link.

When implementing a dropbutton, it is important to pay attention to the order of items in the list, in particular the first item, which will of course be the label of the button, and thus should be the most likely to be used, or at least the most descriptive for the collection of items in the list.

If a dropbutton has only a single item in it, it goes without saying that it won't have its dropmenu (and accompanying arrow).

If it is important for the user to recognize the presence of an item in the dropdown, and there is concern that he or she might overlook it, a separate one-item dropbutton can be placed next to the dropbutton with the individual item. This might be an 'update' item only available under certain circumstances, and which would in all likelihood require action when present.

A feature that needs some usability testing before it can be okay’d is for the dropbuttons, when used in repeating setups, like a list of comments, as on the Comments page, clicking an item in the list (except of course the topmost one), would change all the repeating dropbuttons to use that item as their topmost one. This might help when going through for instance a list of comments and marking individual comments as spam, that it would make the process faster, by adapting the interface to the current task being performed.

Tabs and Pagesplitter

The purpose of the tabs, alongside the pagesplitter, is to hide from view sections of a page which are not essential to the basic operation of the page; thus minimizing the amount of clutter. These include advanced settings, media silos, timelines and the like.

Clicking an inactive tab opens its pagesplitter. Clicking an active tab closes its pagesplitter. Clicking an inactive tab when another tab is active closes the active tab and in turn opens the newly selected tab's pagesplitter.

A tab row can include any number of tabs, of which either none or only a single tab can be active at any given time. Up to an including seven tabs (each tab in a row being 100px wide) will display in a row, and beyond that, in a dropbutton, like so:

Tabs are bound to a Pagesplitter, so called because it 'splits' the page, revealing what is behind it. As only a single tab in a row of tabs can be active at any given time, it goes without saying that only a single Pagesplitter can be active at any given time, per tab row.

Multiple tab rows per page are possible, though it is strongly suggested that tab rows not be placed one after the other, without any other page elements in-between.

It is also strongly suggested that tabs be placed in relation to any other content they might be related to, as in the case of media silos being placed just prior to the content area on the Create pages, or the Timeline being placed after the main navigation area, but before the content area on the Comments page. Also, due to their full-width nature, they need to be placed outside of any other containers.

A design for vertically-resizable pagesplitters has been suggested, but at the time of writing has not seen implementation.

The Timeline

The activity timeline is a combined poor-man's statistics graph as well as an indicator of where in the content-archive the currently viewed items belong.

In its default monthly view, the timeline graph basically consists of a number of 1-pixel wide columns, each representative of a single comment or entry. Thus, if one month has seen 100 comments, and another month saw 200 comments, they would be a 100px and 200px wide respectively, showing one month to have been twice as active as the other.

The 'loupe' is a transparent slider, which functions both as an indicator of the current position in the archives timeline, and at the same time as a control for moving around in those archives as well as setting the amount of items viewed. It is per default 20 pixels wide--thus showing 20 items of content--but can be resized in 20 pixel increments up to 100.

The loupe functions as a handle, which can be dragged and dropped on the slider that is the timeline, allowing the user to easily move his view to an approximate point in time.

Here is how the user can interact with it:

- Clicking and holding in the center of the loupe allows it to be dragged and dropped in 20-pixel increments on the timeline, which in turn will reload the results.
- The loupe can be resized in increments of 20 by dragging its sides.
- Clicking on the timeline, will shift the loupe its own length in the direction of the click.
- Double-clicking the timeline will shift the loupe directly to the double-clicked position.

Note: The following is a feature-suggestion only, pending testing of its implementation viability. Concerns are primarily bandwidth needed and the ensuing loading speeds.

Because each listed item corresponds directly to a 1-pixel-wide column in the timeline, elements on the page can interact with the timeline, to provide the user visually with valuable data. By hovering for instance a name, an IP or an entry name, the related items in the timeline highlight themselves, after which they slowly fade out again.

This allows for the user to hover the name of an entry on the Comments page, and have the density of comments for said entry highlighted on the timeline. On mouseout, the timeline items slowly fade out, allowing the user enough time to realign and/or resize the loupe to take a closer look.

The Tag Weight Graph

Tags themselves have no timestamp, as they are used at will, and so make no sense in a timeline. Tags however have a 'weight', indicative of how many entries they have been used with. On the Tags page, this weight is used to display a timeline-similar graph that can be similarly operated.

Inline Labels

As seen in particular on the Create pages, some form elements have had their label moved into their content area of their target form element. This was done to create as lean and trim a layout as possible, but comes at the cost of possible confusion on the part of the user once those elements have been filled out with user data, and the labels are no longer accessible.

As an aid, elements with inline labels have small 'glue on' labels attached to them on focus and hover.

When using inline labels, it is very important that to do so with the greatest care, and preferably as sparingly as possible. Here are a few guidelines:

- To the extent possible, elements by its size and position, should describe itself.
- The page should be used often enough that the user easily memorizes the inline labels, based on the sizes and positions of the elements.
- The page should preferably contain few, easily distinguishable elements.

Ideal pages for inline labels are the Login page and the Create pages. Using inline labels on options pages and the like is likely to only confuse the user, and should be avoided.

Resize Handles

Some elements, like textareas, benefit greatly from being resizable, to allow the user greater overview and a better use of their vertical (or in some cases horizontal) screen space. Wherever the resize handles are visible next to an element, they can be dragged and dropped, resizing the element in the process.

Sliders

A slider consists of a rail with a drag and droppable handle. Changing the position of the handle on the rail, affects the page in some fashion. One such use in the Habari administration interface is on the Manage pages (Comments, Entries, Pages), where a slider controls the amount of content being shown, from everything through excerpts to nothing.

The slider is a powerful and elegant tool when used correctly, and is particularly helpful when the user is called upon to type in values on a scale, like the hue or brightness of a color, or the size of text.

Livesearch

To speed up the process of filtering results, all search fields in Habari's administration interface use the near-instantaneous 'livesearch', which fetches the results relevant to your query at every keystroke.

Search Filtering

While not truly a custom control, search filtering is a sibling to livesearch, and is therefore included here.

Using a system similar to Google's advanced operators, the search fields in Habari's administration interface function almost as commandlines, for filtering content--particularly on the Manage pages. Instead of having a separate page for filtering out draft entries, a simple search for state:draft is made, and the entries where the state of draft is found, are shown.

These advanced operators provide and extremely powerful method of filtering, which offers the user full control over what is being fetched, and what isn't.

Similarly, for all the applicable links on the Manage pages, like the name of a comment author or a date of publication, actually perform a search, which shows up in the search field, and which is the subsequently editable by the user.

Similarly to Gmail and Google Reader, the hotkey for switching focus to the Search element is '/'.

The full documentation of the search filtering is available elsewhere.

The Dashboard

The dashboard is meant to provide the user with an overview of the current activity on, or relating to, the blog. To achieve this, the dashboard consists of a customizable module system in which the user can freely rearrange, add to or remove modules. Here follows a breakdown of the functionality and operation of the dashboard module system.

Frontend Functionality

The follow operations are possible for the user, preferably without ever reloading the page:

- Rearrange modules by dragging and dropping them.
- View and change options, like the link for a feed, by clicking a module's options icon in the top-right corner of module.
- Remove module, from within its options.
- Select a module-type from a dropdown and add it.

Backend Functionality

The dashboard comes with a default set of basic modules, such as:

- Feed
  - Latest Entries
  - Latest Comments
  - Moderation Queue
- It should be possible to add new modules as standard Habari plugins.
  - Plugin-Modules should show up on the dashboard when activated.

The dashboard comes with a default setup of 'Latest Entries' (latest entries module), 'Latest Comments' (latest comments module) and 'Habari Developer Blog' (feed module).