tychoish, Personal Stack

Release

Sam Kleinman

March 30, 2013

Contents

The goal of the “Stack” project is to collect a number of “quality working defaults,” for programs like Emacs and StumpWM to provides users with a starting point for their own configuration and customization development. While both of these programs (and others) are very powerful tools, without some amount of configuration these tools are not particularly useful.

I’m producing these configurations from the configurations I use every day. The methodology that I used for extracting and packaging this configuration is available in this repository, and

Continue reading for more information about the components of the stack.

– tychoish

1 Releases

1.1 Downloads

Please see the Cyborg Institute download resource for a full listing of available *stack* packages available for download, including archived versions.

Packages

All packages and related tooling are available via git, at all times:

- stack on git.cyborginstitute.net
- stack on github

In general:

- The latest version of stumpwm-config
- The latest version of emacs-config

You can always view the directory listing at downloads.cyborginstitute.net for a full list of archived packages.

Documentation

The latest version of this manual is available for download in ePub and PDF formats:

- Stack Manual, ePub
- Stack Manual, PDF

See more information on "*Downloads*" page.

2 Documentation

2.1 Emacs Stack

Overview

Emacs is an incredibly useful piece of software, but the default configuration is difficult to use, and makes it difficult to appreciate the capabilities of the program, understand how to customize the environment, or even use the software effectively for basic text editing tasks. Once you become familiar with Emacs and have begun to customize it yourself, the rational for the "no default configuration," becomes clear, but it's difficult to get to this point and there's no good reason to leave would-be emacs users to fend for themselves.

The "emacs-stack", then, provides a good example configuration that users may find helpful as an example of how to manage a large and complex emacs configuration, and a set of good, working defaults derived from tycho's working configuration. In contrast to some other attempts to provide a good introductory default to emacs, this "distribution," of emacs code does not attempt to package emacs itself, nor does it attempt to deliver an emacs experience designed to be easy to learn for users of another system or environment. Rather, the emacs-stack presents a faithful example of a real "working" emacs configuration.

The distribution contains a directory named `emacs/` that contains all emacs lisp files and a `makefile` that manages installation, removal, and byte-compilation of the emacs configuration. Most of the files in this distribution are publicly available and freely licensed packages, but there are a number of files that configure key-bindings, configure variables and settings, and provide some custom "glue functions" that are original to this configuration.

Release

The latest release is always available at http://download.cyborginstitute.net/emacs-config.tar.gz.

You can always view past archival releases by visiting http://download.cyborginstitute.net/, where releases reflect the full commit hash of the stack project git repository (github).

Please submit an issue to the mailing list if you'd would like to see a more formal release process or numbering system.

Points of Interest

- `tycho-` files contain specific configuration and integration of the emacs packages into the "stack." Of particular note are `tycho-emacs.el` and `tycho-init.el`: `tycho-emacs.el` contains the core configuration, while `tycho-init.el` guides the initialization process for running multiple instances of emacs' daemon mode with different configurations.
- `tycho-keybinding.el` holds most of the keybindings. Review this file before using emacs-stack to become familiar with the interface.
- `config/` the files in config, provide the "gateway" in the initialization process to all of the `tycho-` files. All the files reflect a hostname for one of your systems, and make it possible have machine specific configuration. The `makefile` will a file from this directory to your systems `~/.emacs` file.
- `makefile` will byte-compile *all* files in the distribution, create a host specific file in the `config/` directory, archive your current `~/.emacs` file, and create a symbolic link to your new emacs config.

Setup and Customization

See the *downloads page* for a link to the latest tarball with a copy of the emacs-stack release. Download this file, using the following command:

```
cd
wget http://download.cyborginstitute.net/emacs-stack.tar.gz
```

Then extract the archive:

```
tar -xxv emacs-stack.tar.gz
```

Then run the `make` process to install the new configuration.

```
make install
```

When complete the script will archive your existing `~/.emacs` file in your home directory and `~/.emacs` will be a symbolic link that points to a file in `emacs/config/`, and all emacs lisp files will be byte-compiled.

You can edit any of the emacs lisp files as ended for your configuration. Run the following make operation at any point after changing the lisp files to update the byte-compiled files:

```
make byte-compile
```

Continue reading for more information on the specific opportunities for customization and components of the emacs-stack.

Internal Overview

Many of the files included in the distribution are third-party libraries and scripts. The files in the `config/` directory, and all files beginning with `tycho-` provide the integration and originate with this distribution.

Note: There is no guarantee that any of the emacs lisp included in this package is: bug free, up to date, or unavailable through other means.

The following list introduces the core components of the emacs stack, that create the entire experience

`config/hostname.el`
: Contains all machine specific configuration. Ideally, these files are all unique to the machine, but there is some duplication in practice.

  Your user account's `~/.emacs` file should be a symlink to this file.

  Do not insert lisp into this file unless it causes an inter-system compatibility issue.

`tycho-init.el`
: This file controls the initialization process, and grows out of a need to maintain two or more emacs daemon instances on the same system *with* different desktop (i.e. state) systems.

`tycho-emacs.el`
: This is the core configuration file, and most of the other `tycho-` files are required from this file. At some point in the distant history all of the `tycho-` files *were* in this file, now `tycho-emacs.el` contains `(require)` calls and sets a number of variables and settings.

`tycho-display.el`
: Contains all visual modifications to emacs' display and font selection. Implemented as a series of functions the `init` file requires this file and then calls one of these functions during the display process.

  Alternatively, you may, use your `~/.Xdefaults` file with the following lines to control your Emacs appearance a bit more cleanly:

```
emacs.menuBar:off
emacs.FontBackend:xft
Emacs.font: inconsolata:pixelsize=14:antialias=true:hinting=true
Emacs.pane.menubar.font: inconsolata:pixelsize=14:antialias=true:hinting=true
```

`tycho-deft.el`
: Provides some fairly significant wrappers around Deft, similar to the code in `tycho-ikiwiki.el`.

`tycho-keybindings.el`
: In most cases, the `tycho-keybindings.el` file specifies all custom keybindings that don't directly relate or depend on other code. For instance org-relgated bindings are stored in `tycho-org.el`.

All other `tycho-` files contain simple wrappers and configuration around otherwise unmodified lisp files and packages obtained from third party sources. Comments may be sparce, but feel free to open an inquiry on lists.cyborginstitue.net for a documentation and/or commenting enhancement.

2.2 StumpWM Stack

Overview

StumpWM is a manual tiling window manager written in Common Lisp, with an interaction paradigm inspired by GNU Screen and Emacs. Stump has a minimal feature set, with support for dual monitors, multiple window groups (i.e. virtual desktops,) and full keyboard driven interaction, and has many opportunities for extension. Although there is minimal current development, the software is stable, has wide use, and is very extensible.

The default StumpWM configuration is operable, however there are no default configurations distributed with StumpWM to provide a "base configuration" for hacking, customization, and practical use. This "StumpWM stack" provides such an example derived from an actual day-to-day StumpWM configuration. In turn, this document, provides an outline of the stack, and its configuration and use.

Release

The latest release is always available at http://download.cyborginstitute.net/stumpwm-config.tar.gz.

You can always view past archival releases by visiting http://download.cyborginstitute.net/, where releases reflect the full commit hash of the stack project git repository (github).

Please submit an issue to the mailing list if you'd would like to see a more formal release process or numbering system.

Use

To use this StumpWM configuration you must:

- have StumpWM installed and operable on your system
- Follow a few simple *steps* to organize the configuration so that StumpWM will read the new configuration.

From here, you can continue to develop and tweak the configuration, and offer your own fork or variant, *or* submit patches/merge (pull) requests to the listserv!

Points of Interest

The distribution of this StumpWM config comes with the following components:

- A `makefile` which contains a default message for "installing" and setting up the StumpWM configuration.

  This process is non-destructive, and will not delete your current StumpWM configuration (if you have one.) Rather, it will move an existing `~/.stumpwmrc` file to a `~/stumpwmrc-` file with a timestamp appended.

- A `stumpwm` directory that holds this StumpWM configuration. This includes components from the `contrib` directory of StumpWM, as well as some semi-custom configuration. Within this directory there are the following components:

  - `config/` this is where the site-specific or host-specific configuration goes. If you have any configuration that you need that depends on the host, or are using the same basic configuration on a number of hosts but have limited per-machine differences, store that configuration here.

    To be fair, you could write Lisp code to implement different configuration subsets based on environment information; however, this is difficult to use and track if you have to add and possibly remove hosts regularly/ever it might become confusing and hard to maintain.

  - `tycho-` files inside of the `stumpwm/` configuration, contain lisp which is mostly unique to this configuration. These files are where all of the "interesting" things happen:

    * `tycho-keybindings.lisp` holds all keybindings.
    * `tycho-system.lisp` holds all custom commands and definitions.
    * `tycho-settings.lisp` holds all configuration and settings that aren't keybindings or functions.

  - Other lisp files, typically derived from the StumpWM contrib folder.

Setup and Customization

To use:

1. Install StumpWM and all of its dependencies. Whatever method you use to install these software packages should be sufficient. The most common configuration for StumpWM is to use: the `sbcl` lisp system, the latest `clx` version (i.e. from git,) and a compiled/installed version of itself StumpWM from git. There are packages for this on Arch Linux. Other distributions, and quicklisp should be functional for this task, but have not been tested.

2. Download the distribution, into your home directory. The command will resemble:

   ```
   cd
   wget http://download.cyborginstitute.net/stumpwm-config-latest.tar.gz
   ```

3. Extract the archive:

   ```
   tar -zxvf stumpwm-config-latest.tar.gz
   ```

4. Build the "configure" target.

   ```
   cd ~/stumpwm
   make configure
   ```

5. Inspect the content of the `~/stumpwm-config/stump` directory. Confirm that the configured stumpwm instance is suitable.

6. Build the "install" target. This will rename and timestamp your current `.stumpwmrc` file and create a new `.stumpwmrc` that is a symbolic link which points to `~/stumpwm/config/[hostname].lisp`, that replaces `[hostname]` with the hostname of your machine.

   ```
   make install
   ```

7. Restart StumpWM with the new config.

Continue to tweak the lisp files in and `~/stumpwm` as well as `[hostname].lisp` as needed. In general, place all machine-specific configuration in `[hostname].lisp`, and general configuration in the `tycho-` or other files in the `~/stumpwm` directory.

Index