

日本タイル型ウィンドウマネージャ推進委員会

日本タイル型ウィンドウマネージャ推進委員会とは

タイル型ウィンドウマネージャの普及を推進する任意団体です。略称は**日本タイル**です。

タイル型ウィンドウマネージャの布教および広報、メーリングリストにおける情報交換、ドキュメントの翻訳などを目的として設立されました。

タイル型ウィンドウマネージャとは？

- ウィンドウ配置はウィンドウマネージャの仕事です
- ウィンドウの制御は主にキーボードで行います
- マウス操作が最小限、または不要になります
- ウィンドウはウィンドウマネージャがタイル状に配置するため、基本的に重なりあうことはありません
- 無駄なスペースが発生せず、画面を有効に活用できます

日本タイルへの参加方法

日本タイルへの参加方法

メーリングリストへの登録をもって参加としています。

<http://lists.sourceforge.jp/mailman/listinfo/tilingwm-users>

タイル型ウィンドウマネージャはこんな人にオススメ

- とにかく軽量なウィンドウマネージャを使いたい
- 画面が狭いので画面を有効に使いたい
- 画面が広いのでマウスでの移動が煩わしい
- いつもウィンドウが重ならないようマウスで再配置
- マルチディスプレイでウィンドウの操作が面倒だ
- GUI 環境でもショートカットだけで生きていきたい
- GNU screen にどっぷりハマってる
- Windows1.0 が好きだった

タイル型ウィンドウマネージャの種類（UNIX 系）

名称	特徴	URL
awesome	dwm をベースにさまざまな機能拡張を施したウィンドウマネージャ。設定ファイルによる高いカスタマイズ性を備える。awesome3 から設定ファイルが Lua に移行。現在人気急上昇中。	http://awesome.naquadah.org/
dwm	シンプルさがウリ。様々な派生タイル型ウィンドウマネージャの元祖。設定ファイルが存在せず、コンパイル前に設定を投入する。	http://www.suckless.org/dwm/
ratpoison	GNU Screen に似た操作性をもつ。その操作性と日本のニュースサイト／ブログ等で何度か紹介されたことから日本ではそこそこ人気。	http://www.nongnu.org/ratpoison/
stumpwm	Ratpoison の後継。Common Lisp で実装されている。Lisper にはいいんじゃないでしょうか。	http://www.nongnu.org/stumpwm/
wmii	独自のウィンドウ分割・制御を採用。dwm と同じ開発元でプログラムソースはとてもシンプル。	http://www.suckless.org/wmii/
xmonad	Haskell で書かれた柔軟性の高いウィンドウマネージャ。XRandR, Xinerama を完全にサポートし、複数ディスプレイ環境でのウィンドウ制御は他のタイル型ウィンドウマネージャの追随を許さない。	http://xmonad.org/

Windows におけるタイル型ウィンドウマネージャのようなソフトウェア

名称	特徴	URL
HashTWM	Windows 用の中では一番タイル型ウィンドウマネージャらしい挙動。	http://dockbox.demonastery.org/
dwm-win32	dwm の移植版。基本的に同じ挙動だがちょっと重い。	http://www.brain-dump.org/projects/dwm-win32/
EIZO ScreenSlicer	さまざまな制御ができるが、EIZO 製品利用者のみ使用可。	http://www.eizo.co.jp/products/lcd/screenslicer/

Mac OS X におけるタイル型ウィンドウマネージャのようなソフトウェア

名称	特徴	URL
TwoUp	画面を左右に 2 分割、ショートカットキーでウィンドウ制御が可能	http://www.irradiatedsoftware.com/twoup/
SizeUp	TwoUp の上位版。ただしシェアウェア。	http://www.irradiatedsoftware.com/sizeup/

詳しくは「日本タイル」で検索！

今日からはじめる awesome チュートリアル (3.4 対応版)

今日からあなたも awesome で快適ウィンドウマネージャ生活！

awesome とは <http://awesome.naquadah.org/>

Debian 開発者でもある Julien Danjou が 2007 年から dwm をベースに開発を開始した tiling window manager(タイル型ウィンドウマネージャ)です。進化に進化を重ね、現在では awesome 自身を *a highly configurable, next generation framework window manager*(高度にカスタマイズ可能な次世代のウィンドウマネージャフレームワーク)と定義しています。

タイル型ウィンドウマネージャの一般的な特徴は日本タイルの Wiki[1]を参照してください。

awesome の主な特徴:

- たくさんの Freedesktop.org 標準を実装。
 - システムトレイに格納されるアプリケーションも問題なく動作。
- XRandR, Xinerama 対応。
 - マルチディスプレイ環境でも問題なし。
- 設定ファイルに Lua を採用。
 - プログラマブルで自身で拡張可能[4]。

インストール編

awesome はその活発な開発のためディストリビューション付属のパッケージは古い場合があります。今から awesome 2.x の使用は上記の利点を失うことになるので、推奨しません。最低でも 3.1 以上の利用を推奨します。awesome 2.x と 3.x では操作体系はかわりませんが、設定ファイルと機能が大幅に異なるため、継続的に使うには 2.x は古すぎます。

Debian

Julien が Debian 開発者ということもあり、彼自身がパッケージングしています。2010 年 9 月現在、Debian におけるバージョンは以下の通りです。

- 安定版 ('lenny'): 2.3.3
- テスト版 ('squeeze'): 3.4.6

Debian GNU/Linux squeeze/testing

テスト版ユーザはパッケージをそのまま使用できます。

```
# aptitude install awesome
```

インストール後、ログアウトするとディスプレイマネージャのセッションで awesome が選択できます。

Debian GNU/Linux lenny/stable

残念なことに、xcb 関連のライブラリが lenny に収録されてる 1.1 と awesome が要求する 1.4 で互換性がないため、全パッケージの再ビルドが必要となり、lenny で awesome 3.4 のパッケージをビルドすることは現実的ではありません。自分でなんとかがんばるか、操作だけでも習得したい場合は非常に残念ですが awesome 2.3.3 を使用してください。

```
# aptitude install awesome
```

Ubuntu

Ubuntu 10.04 では awesome 3.4.3 が使用できますので、インストールは簡単に行えます。

```
# aptitude install awesome
```

インストール後、ログアウトするとディスプレイマネージャのセッションで awesome が選択できます。

Gentoo

Gentoo では最新リリースが Portage ツリーに入っています。

- stable: 3.4.4
- unstable: 3.4.7

```
# emerge awesome
```

インストール後、ログアウトするとディスプレイマネージャのセッションで awesome が選択できます。

awesome 体験編

このチュートリアルを印刷するか、もしくは別のパソコンで開いて、見ながら試してください。VMware や VirtualBox などの仮想化環境で awesome を試すのもいいと思います。本チュートリアルでは、awesome 3.4 に準じます。

Mod4 キー

awesome のキーバインドの prefix は Mod4 キーです。通常、このキーは左側の Windows キー、または Command キーです。この設定は変更できますが、awesome 側は Mod4 のままで、xmodmap を使い、特定のキーを Mod4 に変更するのが得策です。スペースキーの左隣りが無変換キーの人は

```
$ xmodmap -e 'keysym Muhenkan = Super_L'
```

で、無変換キーを左の Windows キーに割当てられます。英字配列の MacBook だと、Command キーがちょうどいい位置にあります。素晴らしいですね。みなさん英字配列にしましょう。

Lesson 1: ターミナルを起動！

まずはターミナルを起動しましょう。ここでは未設定なので、xterm が起動します。

- Mod4-Return (ターミナルを起動する)

ウインドウが最大サイズで現れたと思います。

```
$ date | less
```

date コマンドを実行して時刻を記録しておきましょう。ウインドウがどのように移動したかを追跡します。もう 1 つターミナルを起動しましょう。

- Mod4-Return (ターミナルを起動する)

初めに開いたウインドウがどうなったかを確認してください。ここで、もう 1 度 date | less を実行してください。ウインドウが現れた後、どのウインドウにフォーカスが当たったかを確認してください。新しいウインドウにフォーカスが移動しているはずです。また、ウインドウは縦に分割されたはずです。古いウインドウがどこに移動したかも確認してください。右へ移動したはずです。

しつこいですがもう 1 度ターミナルを起動します。

- Mod4-Return (ターミナルを起動する)

同じく、date | less を実行し、どのようになったか確認してください。一番古いウインドウは右下へ移動しているはずです。一番新しいウインドウは常に左側に 1 つだけ出ること確認してください。この左側の領域を *master* と呼び、対して右側を *slave* と言います。作業領域が一番広いのが master です。通常はこの master 領域で作業します。

awesome の設定を編集しながら基本操作を覚えましょう。

Lesson 2: ウインドウサイズの制御

まず、master のウインドウで q を押し、less から抜けます。しばらくここで作業を行います。とにもかくにも、ターミナルに xterm を使っている人は少数派だと思われるので、そこからカスタマイズします。

デフォルトの設定ファイルは/etc/xdg/awesome/rc.lua です。拡張子からもわかるように、設定ファイルは [Lua](#) を使っています。Lua は軽量でアプリケーションの拡張用言語(アプリケーションそのものに組み込まれる)として広く使われています。デフォルトの設定を自分のホームディレクトリへコピーします。

```
$ mkdir -p ~/.config/awesome
$ cp /etc/xdg/awesome/rc.lua ~/.config/awesome/rc.lua
```

ユーザ用の設定ファイルは~/.config/awesome/rc.lua です。

```
$ vi ~/.config/awesome/rc.lua
```

master の領域が少し狭いと感じた人はウィンドウを広げてみましょう。

- **Mod4-l (master 領域にあるウィンドウのサイズを増加させる)**

1 度押すとウィンドウが広がります。広げすぎた場合は

- **Mod4-h (master 領域にあるウィンドウのサイズを減少させる)**

これがウィンドウサイズを変える方法です。この操作は頻出なので体に覚えさせましょう。マウスを使う方法もあります。

- **Mod4-右クリック+ドラッグ (フォーカスのあるウィンドウのサイズを増減させる)**

適当なサイズにできたら、編集に戻ります。まず 12 行目の変数 `terminal` を修正しましょう。私は `"mlterm --sb=false"` にしています。Ubuntu 系の人は `"gnome-terminal"` などがいいでしょう。

修正したら、`awesome` を再起動します。

- **上部一番左の a マークをクリック→awesome→restart**

再起動するとウィンドウサイズやフォーカスが元に戻ります。ターミナルも再設定したことなので、一旦すべて閉じましょう。master の `vi` を終了 してください。ターミナルであれば、`exit` や `C-d` で閉じれますが、ここでは `awesome` 的な閉じ方を学びます。

- **Mod4-Shift-c (フォーカスのあるウィンドウを終了する)**

アプリケーションは基本的にこのキーバインドで閉じます。3つのウィンドウすべてを同様に閉じましょう。改めてターミナルを起動します。ここからは練習をかねて `awesome` 内で Firefox を立ち上げ、このページを見ながら操作してみましょう。このページのアドレスは <http://tinyurl.com/awesome-tutorial> になります。

`awesome 3.x` からはランチャが内蔵されているので、

- **Mod4-r (awesome 内蔵のランチャを起動する)**

上部のタブ一覧の右に `Run:` というプロンプトが出ていることを確認してください。ここで `firefox` を入力し、`Return` を押します。(ちなみに、`tab` で補完もできます)

……では、`awesome` 内の Firefox からこのページを再び開いていただけたと思いますので、再開しましょう :-)。 (ただし、説明の都合上、最後まで紙などを併用してください…)

Lesson 3: タグを使う

上部左に数字が並んでいるのを確認してください。1 から 9 の 9 つのタグがあらかじめ用意されています。タグは Gnome や KDE で言うワークスペースです。ワークスペースと異なるのはそれぞれがマージできるという点です。たとえば、タグ 1 番と 3 番を表示する、またはタグ 1、2、3、4 を表示するなどです。

ブラウザは大きく見たい時と、ブラウザを参照しながら master で作業をしたい時のどちらにも対応できるように、ブラウザ専用のタグを決めます。ここでは、2 番のタグをブラウザ専用とします。

今のタグは 1 番であることを確認した上で、

- Mod4-Shift-2 (フォーカスのあるウィンドウを 2 番のタグへ移動する)

Firefox が消え、ウィンドウがなにもなくなったことを確認してください。Firefox がきちんと 2 番のタグに移動したかを確認しましょう。

- Mod4-2 (2 番のタグへ移動する)

Firefox が現われたはずです。ほっとしたところで、もとのタグへ戻りましょう。

- Mod4-Esc (ひとつ前のタグへ移動する) または
- Mod4-1 (1 番のタグへ移動する)

どちらも結果は同じです。そろそろ、キーバインドを追加したくてうずうずしている人はここで設定してみましょう。ターミナルを起動します。

- Mod4-Return (ターミナルを起動する)

さて、ここからブラウザを見ながら作業をしたいと思います。この機能はとても重要なので体に覚えさせましょう。

- Mod4-Ctrl-2 (今のタグと 2 番のタグを一時的にマージ(合併)する)

ブラウザがあらわれたはずです。フォーカスを移動する手段をまだ伝えていませんでした。フォーカスの移動は

- Mod4-j (次のウィンドウへフォーカスを移動する)
- Mod4-k (前のウィンドウへフォーカスを移動する)

Firefox とターミナル間のフォーカスがきちんと移動しているか確認してください。フォーカスをターミナルにして、設定ファイルを開きます。

```
$ vi ~/.config/awesome/rc.lua
```

設定ファイルについては、[Wiki](#)[4]にまっています。各用語について定義されているので、一通り目を通してください。キーバインドは API (awesome の基本 API は awful) の awful.key 関数を介して設定します。見ればなんとなくわかるようになります。例えば、これまでに使用したキーバインドは以下のように定義されています。

```
awful.key({ modkey,           }, "Return", function () awful.util.spawn(terminal) end),
awful.key({ modkey,           }, "l",      function () awful.tag.incmwfact( 0.05) end),
awful.key({ modkey,           }, "h",      function () awful.tag.incmwfact(-0.05) end),
awful.key({ modkey, "Shift"   }, "c",      function (c) c:kill() end),
```

第1引数が修飾キーで、第2引数がそれに対応するキーです。そして、第3引数が実行する関数です。基本的な操作は `awful` というライブラリにまとめられています。特定のコマンドを起動させたい場合は `awful.util.spawn(command)` を使います。1 つ例を挙げると、

```
awful.key({ modkey, "Control" }, "m", function ()
    awful.util.spawn("/home/nabeken/bin/twitter-update.sh") end),
```

これを設定すると `Mod4+Ctrl+m` で今 `amarok` で聞いている音楽を `twitter` へポストしてくれます[2]。

Lesson 4: ウィンドウの移動

ここまでで、たどたどしいですが、なんとか使えるようになったはずです。しかし、あと1つ教えなければならぬことがあります。それはウィンドウの入れ替えです。これにはいくつか方法があります。今、ターミナルが `master` とします。Firefox を `master` にしたい時は、まずフォーカスを Firefox へ合せます。

- `Mod4-j` (次のウィンドウへフォーカスを移動する)
- `Mod4-k` (前のウィンドウへフォーカスを移動する)

Firefox 上で、

- `Mod4-Ctrl-Return` (フォーカスのあるウィンドウを `master` へ移動する)

これで Firefox が `master` に移動したはずです。フォーカスも Firefox にあることを確認してください。もう1つの方法は手動で入れ替える方法です。わかりやすくするために、もう1つターミナルを起動させます。

- `Mod4-Return` (ターミナルを起動する)

ウィンドウが3つになり、新たに起動したターミナルが `master` に来たはずです。ここで

- `Mod4-Shift-j` (フォーカスのあるウィンドウを次のウィンドウと入れ替える)

をすると、ターミナルと Firefox が入れ替わったと思います。そして、フォーカスはターミナルのままのはずです。もう1度するとどうなるでしょうか。

- `Mod4-Shift-j` (フォーカスのあるウィンドウを次のウィンドウと入れ替える)

次はそのとなりのターミナルと入れ替わり、新たに起動したターミナルは右下に移動したはずです。もう1度するとどうなるか考えてみましょう。最初の位置、つまり `master` へ戻ります。これと逆方向の操作が、

- `Mod4-Shift-k` (フォーカスのあるウィンドウを前のウィンドウと入れ替える)

結果はそれぞれで試してみてください。これでとりあえず使うには問題ないはずです。今日の操作をまとめましょう。

Lesson 5: Tutorial 1 のまとめ

- Mod4-Return (ターミナルを起動する)
- Mod4-Shift-c (フォーカスのあるウインドウを終了する)
- Mod4-j (次のウインドウへフォーカスを移動する)
- Mod4-k (前のウインドウへフォーカスを移動する)
- Mod4-Ctrl-Return (フォーカスのあるウインドウを master へ移動する)
- Mod4-h (master 領域にあるウインドウのサイズを減少させる)
- Mod4-l (master 領域にあるウインドウのサイズを増加させる)
- Mod4-1 (1 番のタグへ移動する)
- Mod4-2 (2 番のタグへ移動する)
- Mod4-n (n 番のタグへ移動する)
- Mod4-9 (9 番のタグへ移動する)
- Mod4-Ctrl-2 (今のタグと 2 番のタグを一時的にマージ(合併)する)
- Mod4-Esc (ひとつ前のタグへ移動する) または
- Mod4-Shift-2 (フォーカスのあるウインドウを 2 番のタグへ移動する)
- Mod4-Shift-j (フォーカスのあるウインドウを次のウインドウと入れ替える)
- Mod4-Shift-k (フォーカスのあるウインドウを前のウインドウと入れ替える)
- Mod4-F1 (awesome 内蔵のランチャを起動する)

他のチュートリアルとしては

- awesome ちゅーとりある。あるいはタイル型ウィンドウマネージャの勧めのようなもの[3]

が簡単な例とともに画像付きで解説されています。設定ファイルは変わっていますが、キーバインドは変わっていません。こちらも併せてどうぞ。

Extra Lesson: Awesome を拡張する

Awesome の醍醐味はその拡張性にあります。例えば画面上部のステータスバーやタイトルバーを拡張し、CPU 使用率やメモリ使用率を表示することができます[5, 6, 7]。本紙面ではこれ以上紹介できませんが、参考文献を参考に拡張に手を出してみたいかがでしょう。Awesome ならできます。

参考文献

- [1] <http://sourceforge.jp/projects/tilingwm/wiki/FrontPage>
- [2] <http://blog.guillermoamaral.com/2007/03/18/twitter-amarok-update/>
- [3] <http://d.hatena.ne.jp/yaske/20071009/p1>
- [4] http://awesome.naquadah.org/wiki/Awesome_3_configuration
- [5] http://awesome.naquadah.org/wiki/Widgets_in_awesome
- [6] <http://awesome.naquadah.org/wiki/Vicious>
- [7] http://awesome.naquadah.org/wiki/User_Contributed_Widgets