

What is the next feature?

...

Who makes next feature of Vim

:echo \$USER

Name: Yasuhiro Matsumoto

Handle: mattn (GitHub: mattn, Twitter: mattn_jp), Vim hacker.

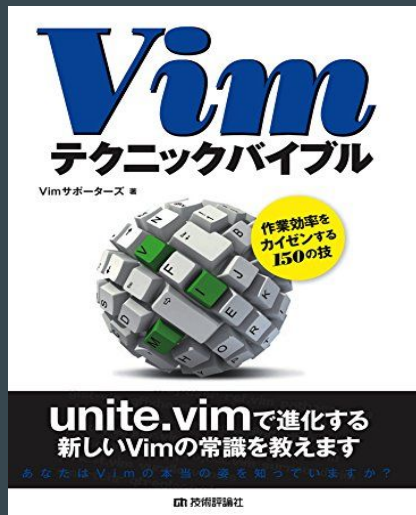
Job: Software Engineer

Skills: Vim, Go, C/C++, C#, Java, Perl, etc...

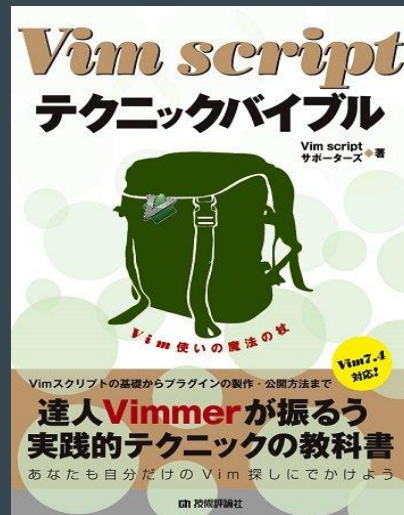
Captain of vim-jp. Number of GitHub repositories: over 1.4k, webapi-vim, gist-vim, emmet-vim, anko, go-sqlite3, go-oci8, etc...

Mainly makes software about Vim or Go.

:e ~/Books/Vim/

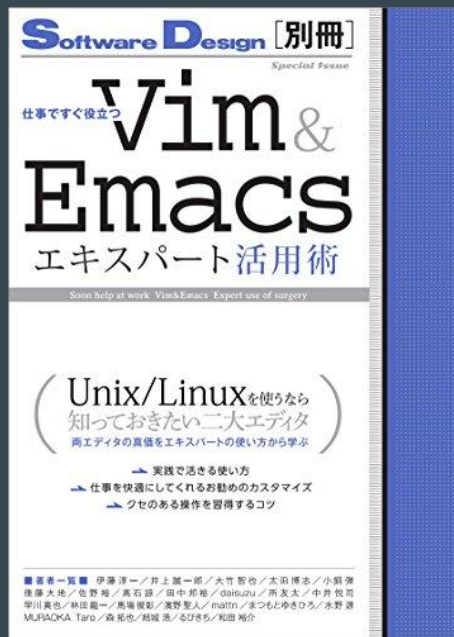


Co-Authors: taku_o, yukimi, thinca,
fuenor, Shougo, mattn



Co-Authors: Shougo, thinca,
kaoriya, mattn

:e ~/Books/Vim/



伊藤淳一, 井上誠一郎, 大竹智也,
太田博志, 小飼弾, 後藤大地, 佐野
裕, 高石諒, 田中邦裕, daisuzu, 所
友太, 中井悦司, 早川真也, 林田龍
一, 馬場俊彰, 濱野聖人, mattn, ま
つもとゆきひろ, 水野源,
MURAOKA Taro, 森拓也, 結城浩,
るびきち, 和田裕介

:e ~/Books/Go/

Yes, I'm Gopher♥



松本雅幸

mattn

藤原俊一郎

中島大一

牧大輔

鈴木健太

[:e ~/Books/Vim-Articles/](#)

Monthly articles of magazine



SoftwareDesign
『Vim の細道』2015/10 ~

:e ~/Books/Vim-Articles/



SoftwareDesign
『Vim の細道』2015/10 ~

`:help agenda`

- Why we started vim-jp
- What is vim-jp
- Who know the next feature of Vim

`:help agenda`

- Why we started vim-jp
- What is vim-jp
- Who know the next feature of Vim

:history

Vi is my first text editor. I used to use vi on everywhere. SONY NEWS, HP-UX, SPARCstation, Linux, BSD(s) and other UNIX OS(s), MS-DOS, and Windows.

After I met Vim, I pulled latest code from CVS every day, and read the diffs.

:history

My first patch to the Vim was

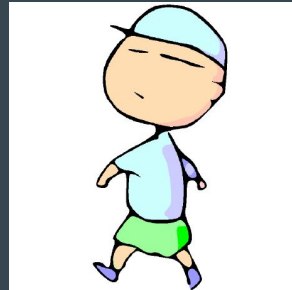
5.6.004 (Improve of Windows IME).

In this time, I didn't know KoRoN(Taro Muraoka) yet. He wrote patches about multibyte support of regular expression for Vim.

:echo "What is Contributing?"

I and Taro were often talking on chat:

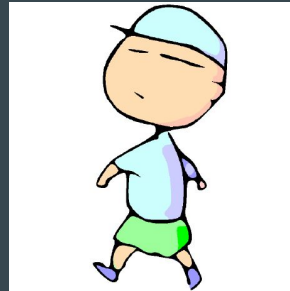
“Why many programmers start a forked project from vi or Vim?”



:echo "What is Contributing?"

I and Taro were often talking on chat:

“Why they didn’t send their patches to the official project?
To improve software, they had to send patches.”



:echo "What is Contributing?"

Some Japanese people (include me) sent patches to vim-dev individually without communicating with each other. So we didn't know:

- Who got trouble?
- What should be fixed?
- Who is writing patches for a problem now?

:echo "What is Contributing?"

Some of Vim communities in Japan (include study group) did good activities about Vim individually. So there were many experts of Vim in Japan, but they often performed duplicated works without sharing information.

What a waste!

:new vim-jp.org

So Muraoka and I started vim-jp in 2011.

vim-jp » Vimのユーザーと開発者を結ぶコミュニティサイト

NEWS: VimConf 2018 が開催されます



vim-jp はテキストエディタ Vim と日本・日本語に関わるあらゆるリソースを集約することを目的としたコミュニティです。

Vim と vim-jp についての詳細はコチラをご覧ください。

[ツイート](#) [Bl 0](#) [いいね! 1](#) [G+](#)

ブログ

2018/07/04 » [The author of Vim will come to VimConf2018](#) [11 users](#)

今年の VimConf 2018 は凄じそ 2013年から毎年開催されている Vim の国際カンファレンス VimConf も今年で 6 回目の開催となります。今年も 11/24、富士ソフトア...

2018/07/03 » [\[攻略記事\] VimConf 2018のスピーカーになるには](#) [2 users](#)

こんにちは、ujihisaです。つい先日 VimConf 2018 CFPがはじまりました。本記事は、応募する気の方、応募するかしないか迷っている方、あるいはすでに応募した方向けの便利情報です。...

2018/06/28 » [VimConf 2018のスピーカーの公募がはじまりました](#) [1 user](#)

Vimを始めよう

バイナリダウンロード

Windows 32ビット
(8.1.0005, +kaoriya, 16.8MB ZIP)

Windows 64ビット
(8.1.0005, +kaoriya, 17.5MB ZIP)

OS X 10.9+
(8.0.1522, +macvim-kaoriya, 14MB DMG)

日本語マニュアル

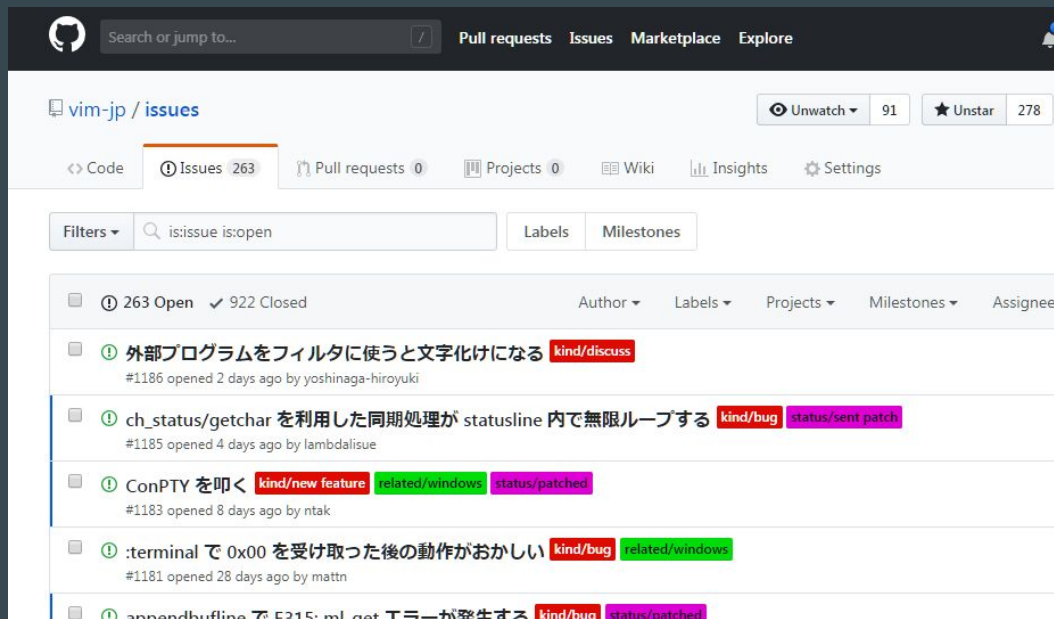
Google カスタム検索

検索もできます。

<https://vim-jp.org/>

:new vim-jp.org

1185 issues, sent 526 patches to vim-dev.



The screenshot shows the GitHub interface for the repository `vim-jp`, specifically the `issues` tab. The page displays a list of 263 open issues. The search filter is set to `is:issue is:open`. The issues are sorted by date, with the most recent at the top. The first issue is titled "外部プログラムをフィルタに使うと文字化けになる" (Using external programs in filters causes character encoding issues), opened 2 days ago by `yoshinaga-hiroyuki`. The second issue, which is highlighted, is titled "ch_status/getchar を利用した同期処理が statusline 内で無限ループする" (Synchronous processing using `ch_status/getchar` causes an infinite loop in the statusline), opened 4 days ago by `lambdalisue`. This issue has labels `kind/bug` and `status/sent patch`. Other visible issues include "ConPTY を叩く" (Using ConPTY) and ".terminal で 0x00 を受け取った後の動作がおかしい" (Action is strange after receiving 0x00 in .terminal).

vim-jp / issues

Unwatch 91 Unstar 278

Code Issues 263 Pull requests 0 Projects 0 Wiki Insights Settings

Filters is:issue is:open Labels Milestones

263 Open 922 Closed Author Labels Projects Milestones Assignee

- 外部プログラムをフィルタに使うと文字化けになる `kind/discuss`
#1186 opened 2 days ago by yoshinaga-hiroyuki
- ch_status/getchar を利用した同期処理が statusline 内で無限ループする `kind/bug` `status/sent patch`
#1185 opened 4 days ago by lambdalisue
- ConPTY を叩く `kind/new feature` `related/windows` `status/patched`
#1183 opened 8 days ago by ntaK
- .terminal で 0x00 を受け取った後の動作がおかしい `kind/bug` `related/windows`
#1181 opened 28 days ago by mattn
- appendbufline で E315: ml_get エラーが発生する `kind/bug` `status/patched`

:new vim-jp.org

vim-jp is Japanese Users Group of Vim

The main purpose of vim-jp is to centralize knowledges about Vim. For example: issues, patches, community events, etc.

:new vim-jp.org

Many OSS user groups refer vim-jp as their ideal model.

`:new vim-jp.org`

Why?

:new vim-jp.org

Because vim-jp have been doing following methods:

- Receive bug reports from Japanese users
 - Confirm reproducibly and figure out cause of bug.
- Write patches to fix bugs
- Document Translation
- Add new features

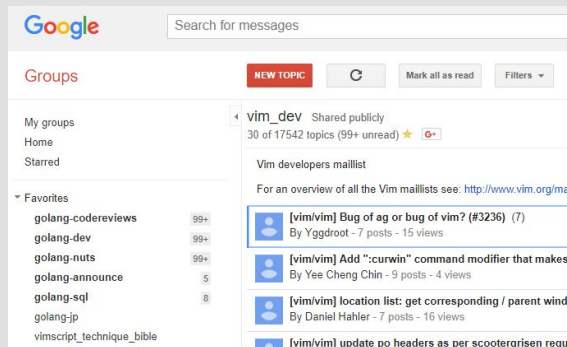
`:help agenda`

- Why we started vim-jp
- What is vim-jp
- Who know the next feature of Vim

`:help what-vim-jp-is`

Bug Report

:help what-vim-jp-is

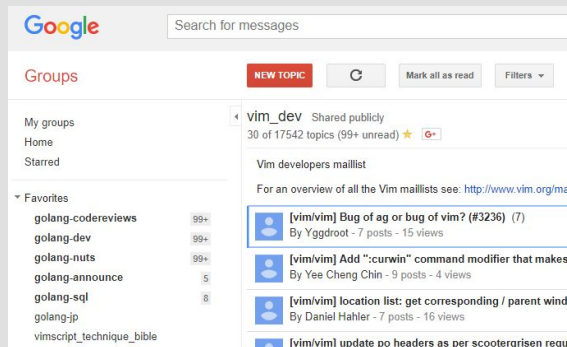


Oh My
English...

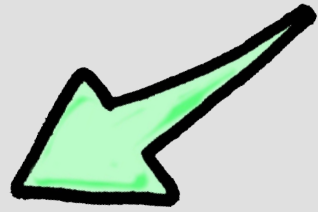


Many Japanese Vim users were giving up reporting bugs since they can't write English well.

:help what-vim-jp-is

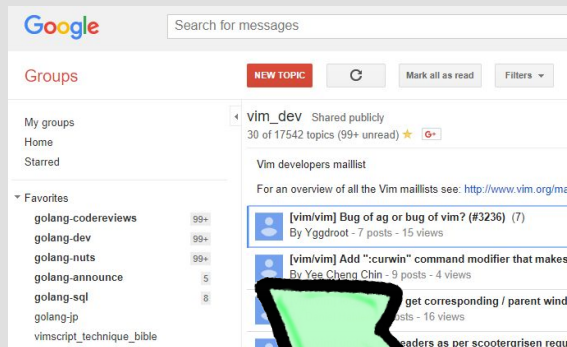


バグ報告



vim-jp

:help what-vim-jp-is



バグ報告

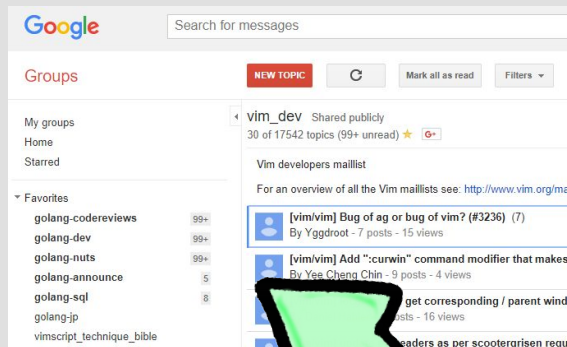


Bug Report



vim-jp

:help what-vim-jp-is



バグ報告



I wrote a patch

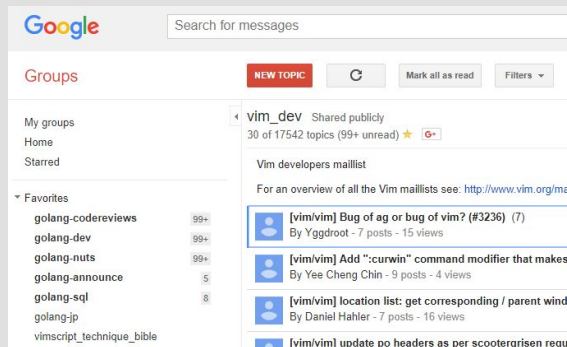


vim-jp

```
:help what-vim-jp-is
```

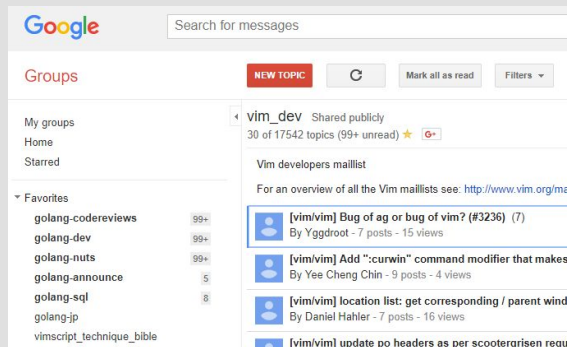
Code Review

:help what-vim-jp-is

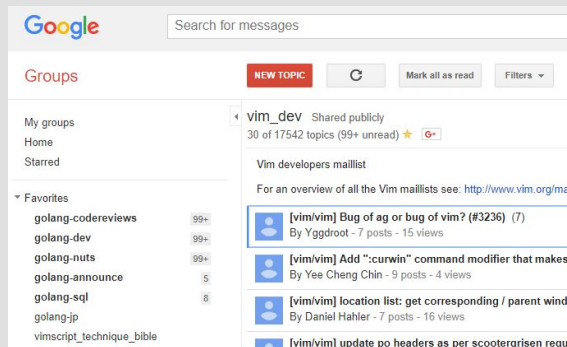


Some Japanese contributors of Vim sent patches to vim-dev each them. i.e. no discussions, no reviews.

:help what-vim-jp-is



:help what-vim-jp-is



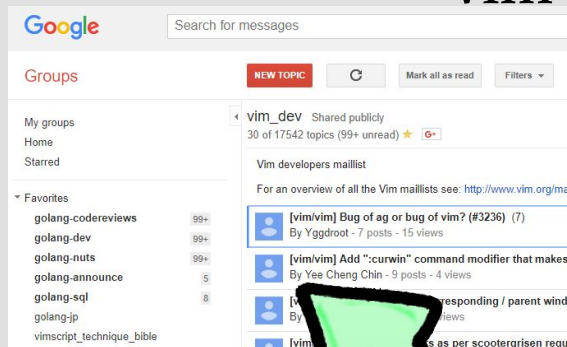
Code Review



vim-jp

:help what-vim-jp-is

vim-jp can send patches on behalf of them.



He wrote a patch



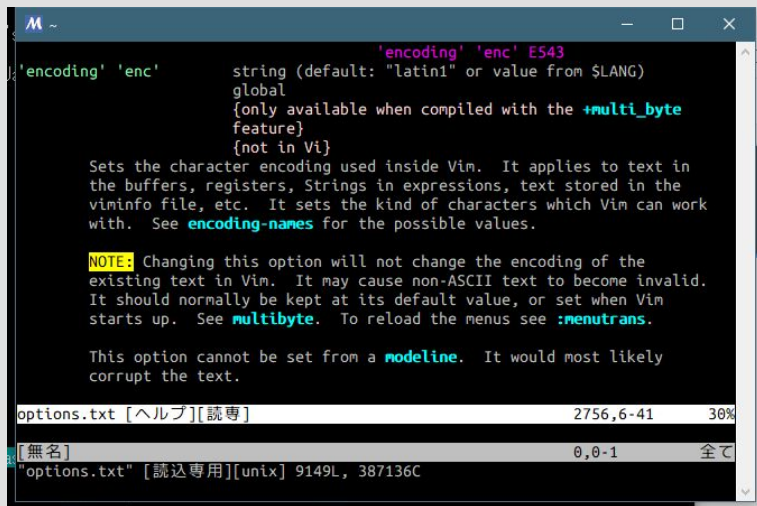
vim-jp

`:help what-vim-jp-is`

Document Translation

:help what-vim-jp-is

Many Japanese people thought reading English help file was difficult.



```
vim -
'encoding' 'enc'      string (default: "latin1" or value from $LANG)
                      global
                      {only available when compiled with the +multi_byte
                      feature}
                      {not in Vi}
Sets the character encoding used inside Vim. It applies to text in
the buffers, registers, Strings in expressions, text stored in the
viminfo file, etc. It sets the kind of characters which Vim can work
with. See encoding-names for the possible values.

NOTE: Changing this option will not change the encoding of the
existing text in Vim. It may cause non-ASCII text to become invalid.
It should normally be kept at its default value, or set when Vim
starts up. See multibyte. To reload the menus see :menutrans.

This option cannot be set from a modeline. It would most likely
corrupt the text.

options.txt [ヘルプ][読専]                2756,6-41    30%
[無名]                                     0,0-1        全て
"options.txt" [読込専用][unix] 9149L, 387136C
```

Oh My
English...



:help what-vim-jp-is

Japanese Translation

vim-jp provide
Japanese version of help files.



vim-jp



:help what-vim-jp-is

Japanese Translation



Help files are translated by many contributors. We, vim-jp, appreciate those contributions.

<https://github.com/vim-jp/vimdoc-ja-working>

vim-jp

`:help what-vim-jp-is`

Add new features

:help what-vim-jp-is

vim-jp added:

:help what-vim-jp-is

vim-jp added:

- Lambda

`:help what-vim-jp-is`

Do you use Lambda?

:help what-vim-jp-is

```
:echo  
 \ "Su Mo Tu We Th Fr Sa\n"  
 \ .join(split(  
 \   repeat(' ', strftime('%w', localtime()) - (strftime('%d', localtime())-1)*60*60*24))  
 \ .join(map(range(1,  
 \   call(  
 \     {y,m->  
 \       [31, 28, 31, 30, 31, 30, 31, 31, 30, 31, 30, 31][m-1] + ((m == 2 && y % 4 == 0 && (y % 100 != 0 || y % 400 == 0)) ? 1 : 0)  
 \     }, [strftime(strftime('%y', localtime())), strftime(strftime('%m', localtime()))]  
 \   )), {_,x->printf('%02d', x)}), ' '), repeat(':', 21).\zs'), "\n")
```

:help what-vim-jp-is

Lambda is very easy!

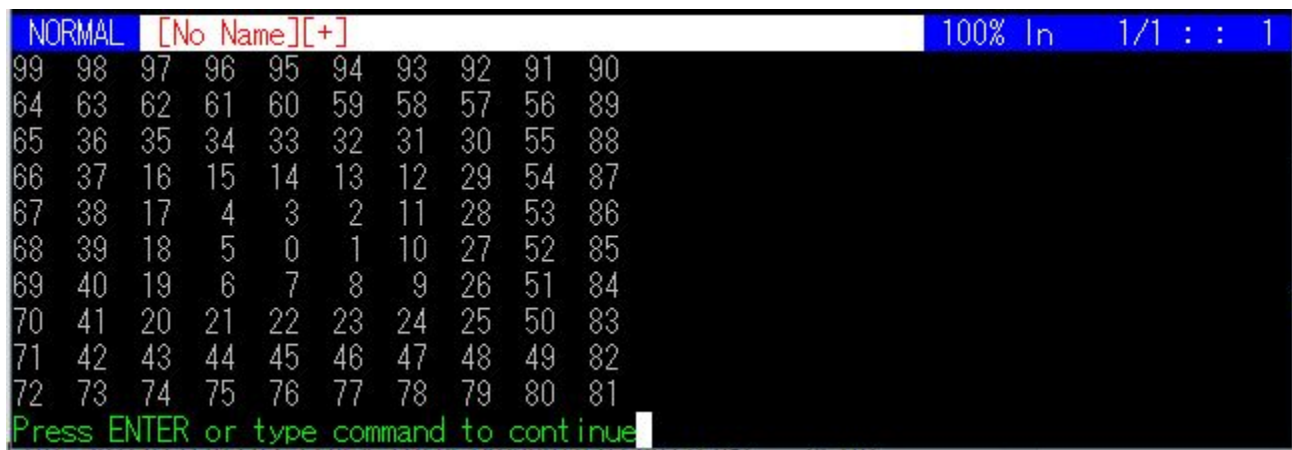


```
NORMAL [No Name][+] 100% In 1/1 : : 1
Su Mo Tu We Th Fr Sa
      01 02
03 04 05 06 07 08 09
10 11 12 13 14 15 16
17 18 19 20 21 22 23
24 25 26 27 28 29 30
31
Press ENTER or type command to continue
```

Thank you @k-takata

:help what-vim-jp-is

Lambda is very easy!



The screenshot shows a Vim editor window with a black background and white text. The status bar at the top is blue and contains the text "NORMAL [No Name][+] 100% ln 1/1 :: 1". The main area of the window displays a grid of numbers arranged in 10 rows and 10 columns. The numbers are as follows:

99	98	97	96	95	94	93	92	91	90
64	63	62	61	60	59	58	57	56	89
65	36	35	34	33	32	31	30	55	88
66	37	16	15	14	13	12	29	54	87
67	38	17	4	3	2	11	28	53	86
68	39	18	5	0	1	10	27	52	85
69	40	19	6	7	8	9	26	51	84
70	41	20	21	22	23	24	25	50	83
71	42	43	44	45	46	47	48	49	82
72	73	74	75	76	77	78	79	80	81

At the bottom of the window, there is a green prompt that reads "Press ENTER or type command to continue".

Thank you @k-takata

:help what-vim-jp-is

```
echo join(call({n->map(range(n),{i,x->join(map(range(i*n,i*n+n-1),{->printf("%*i",float2nr(ceil(log10(n*n-1))),call(function({...->a:1(a:1,a:2,a:3,a:4)},[{e,n,x,y->n%2==0?(y==0?n*n-1-x:(x==n-1?n*n-n-y:e(e,n-1,x,y-1))):(y==n-1?n*n-n+x:(x==0?n*n-n-(n-1)+y:e(e,n-1,x-1,y))))}]),[n,v:val%n,v:val/n]))}),' ')}},[10]),"\n")
```

:help what-vim-jp-is

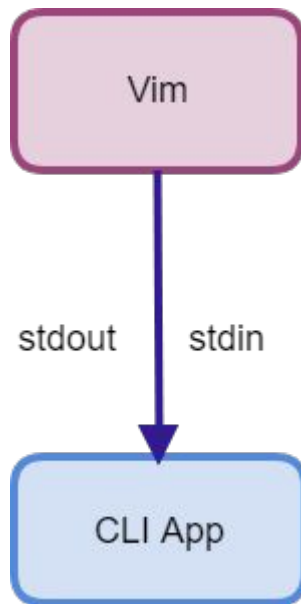
vim-jp added:

- Lambda
- Improvement of job/channel

`:help what-vim-jp-is`

Legacy plugin

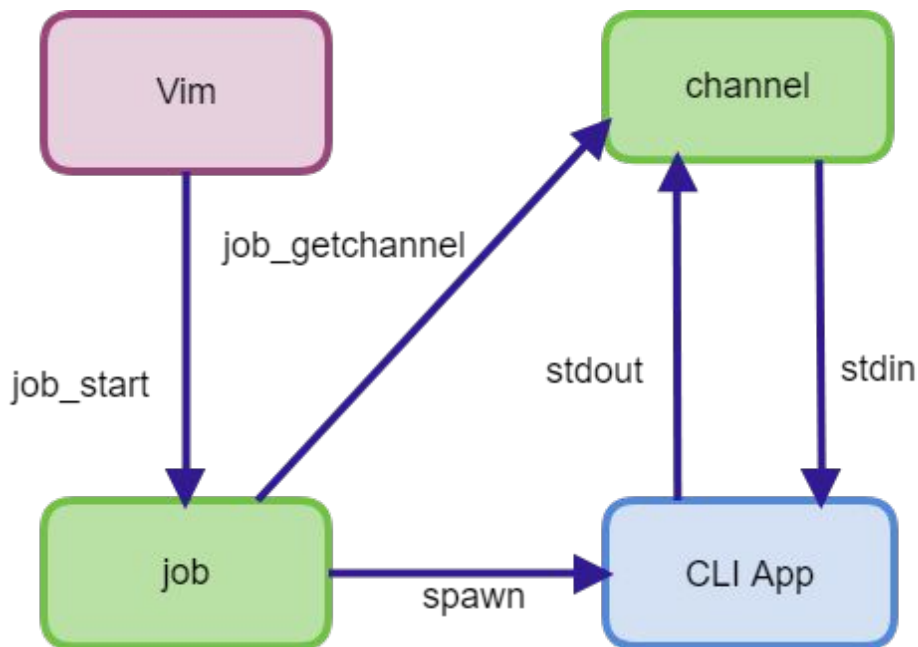
Calling `system()` block Vim
until CLI app exit.



`:help what-vim-jp-is`

Modern plugin

Job/Channel doesn't
block user operation.



`:help what-vim-jp-is`

- `tsuquyomi` (completion for TypeScript)
- `vim-quickrun` (run script quickly in Vim)
- `vim-slumlord` (PlantUML previewer)
- `omnisharp.vim` (completion for C#)

etc...

:help what-vim-jp-is

vim-jp added:

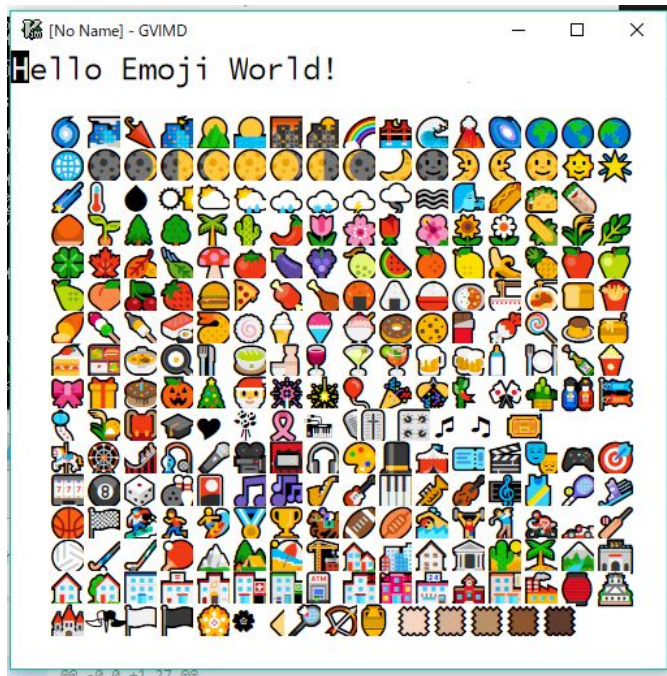
- Lambda
- Improvement of job/channel
- DirectX rendering and color emoji

:help what-vim-jp-is

Do you love beer? 🍺


Thank you @kaoriya, @k-takata





:help what-vim-jp-is






:help what-vim-jp-is

Color Emoji (DirectX support improvements) #2375

 Closed koron wants to merge 74 commits into `vim:master` from `koron:color-emoji`

 Conversation 4  Commits 74  Checks 0  Files changed 6

 koron commented on 26 Nov 2017  

This patch introduces two big features and one small fix to Vim's DirectX rendering.

1. Support colored emoji (emoticon).
2. Improve drawing speed extremely.
3. 'taamode' option didn't work.

Emoji was started from Japan and now widely used in the World as you know. It has become very important parts for the communication. Recent Windows supports colored emoji with DirectX, but Vim couldn't use it. This patch implements colored emoji for Vim.

:help what-vim-jp-is

vim-jp added:

- Lambda
- Improvement of job/channel
- DirectX rendering and color emoji
- incsearch

:help what-vim-jp-is

```
vim - main help file

Move around: Use the cursor keys, or "h" to go left,          h  l
              "j" to go down, "k" to go up, "l" to go right.    j
Lose this window: Use ":q<Enter>".
Get out of Vim: Use ":qa!<Enter>" (careful, all changes are lost!).

Jump to a subject: Position the cursor on a tag (e.g. bars) and hit CTRL-].
With the mouse:   ":set mouse=a" to enable the mouse (in xterm or GUI).
                  Double-click the left mouse button on a tag, e.g. bars.
Jump back:        Type CTRL-T or CTRL-O. Repeat to go further back.

Get specific help: It is possible to go directly to whatever you want help
                   on, by giving an argument to the :help command.
                   Prepend something to specify the context: help-context

                   WHAT          PREPEND  EXAMPLE
                   Normal mode command      :help x
                   Visual mode command      v_      :help v_u
                   Insert mode command      i_      :help i_<Esc>
                   Command-line command      :       :help :quit
                   Command-line editing command c_      :help c_<Del>
                   Vim command argument    -       :help -r
                   Option                  '       :help 'textwidth'
```

Thank you @haya14busa

:help what-vim-jp-is

vim-jp added:

- Lambda
- Improvement of job/channel
- DirectX rendering and color emoji
- incsearch
- :terminal on Windows

:help what-vim-jp-is

```
mbyte.c (c:\dev\vim\src) - GVIM
Microsoft Windows [Version 10.0.17763.55]
(c) 2018 Microsoft Corporation. All rights reserved.

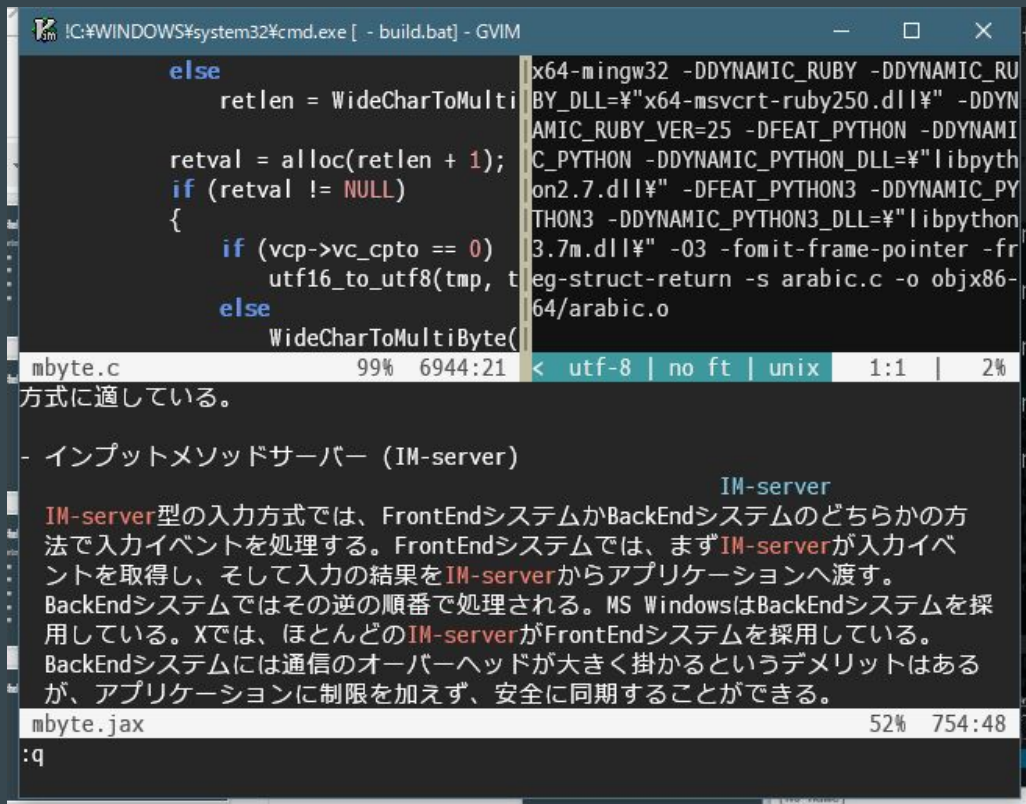
c:\dev\vim\src>

cmd.exe [running] 25% 1:1
    else
        retlen = WideCharToMultiByte(vcp->vc_cpto, 0,
                                     tmp, tmp_len, 0, 0, 0, 0);
        retval = alloc(retlen + 1);
        if (retval != NULL)
        {
            if (vcp->vc_cpto == 0)
                utf16_to_utf8(tmp, tmp_len, retval);
            else
                WideCharToMultiByte(vcp->vc_cpto, 0,

```

N blob | mbyte.c | utf-8 | c | unix 6944:21 | 99%

:help what-vim-jp-is



The screenshot shows a Vim editor window with the following content:

```
else
    retlen = WideCharToMultiByte(
        CP_UTF8, 0, wstr, wcslen(wstr),
        retval, retlen, 0, 0);
    if (retval != NULL)
    {
        if (vcp->vc_cpto == 0)
            utf16_to_utf8(tmp, retval);
        else
            WideCharToMultiByte(
                CP_UTF8, 0, wstr, wcslen(wstr),
                retval, retlen, 0, 0);
    }
}

mbyte.c 99% 6944:21 < utf-8 | no ft | unix 1:1 | 2%
```

方式に適している。

- インพุットメソッドサーバー (IM-server)

IM-server

IM-server型の入力方式では、FrontEndシステムかBackEndシステムのどちらかの方法で入力イベントを処理する。FrontEndシステムでは、まず**IM-server**が入力イベントを取得し、そして入力の結果を**IM-server**からアプリケーションへ渡す。BackEndシステムではその逆の順番で処理される。MS WindowsはBackEndシステムを採用している。Xでは、ほとんどの**IM-server**がFrontEndシステムを採用している。BackEndシステムには通信のオーバーヘッドが大きく掛かるというデメリットはあるが、アプリケーションに制限を加えず、安全に同期することができる。

```
mbyte.jax 52% 754:48
:q
```

:help agenda

- Why we started vim-jp
- What vim-jp doing
- Who know the next feature of Vim

`:help what-vim-jp-is`

Maybe you have realized now.



`:help what-vim-jp-is`

vim-jp is Japanese User Group of Vim



`:help what-vim-jp-is`

vim-jp is Japanese User Group of Vim



`:help what-vim-jp-is`

vim-jp is

Japanese Developers Group of Vim

`:help what-vim-jp-is`

We makes next feature of Vim!?



```
:echo has("new-feature")
```

Hey, vim-jp.
So what is coming next?




```
:echo has("new-feature")
```

Sorry, We don't know.



```
:echo has("new-feature")
```

We are not fortune teller.

But we can bring features by writing code!



```
:echo has("new-feature")
```

Suggestion 1

Support DRCS Sixel on :terminal

`:echo has("new-feature")`

Suggestion 1: Support DRCS Sixel on `:terminal`

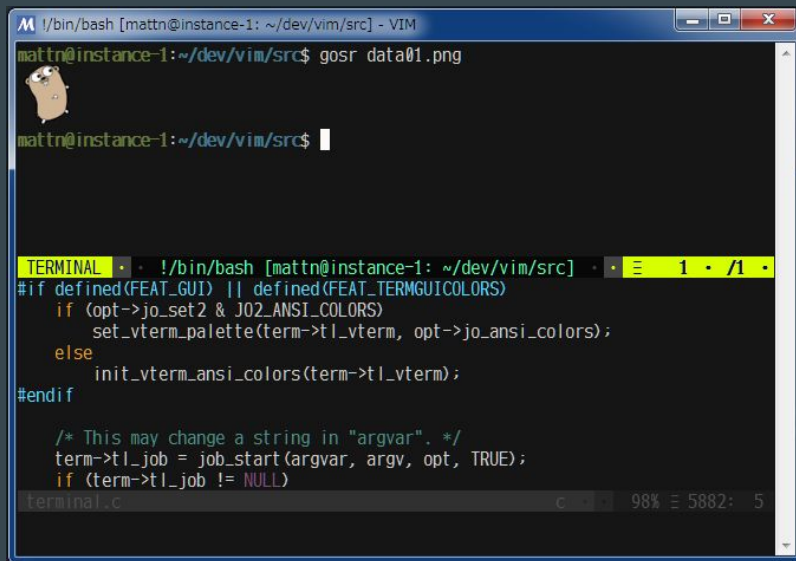
DRCS Sixel is a bitmap graphics format supported by terminals. It can output graphical images on the terminals. But `libvterm` which `:terminal` use does not pass DRCS sequences. So...

```
:echo has("new-feature")
```

Suggestion 1: Support DRCS Sixel on :terminal

Okay, Let's write code!

:echo has("new-feature")



```
! /bin/bash [mattn@instance-1: ~/dev/vim/src] - VIM
mattn@instance-1:~/dev/vim/src$ gosr data01.png
mattn@instance-1:~/dev/vim/src$

TERMINAL · · !/bin/bash [mattn@instance-1: ~/dev/vim/src] · · 1 · /1 ·
#if defined(FEAT_GUI) || defined(FEAT_TERMGUICOLORS)
  if (opt->jo_set2 & JO2_ANSI_COLORS)
    set_vterm_palette(term->t1_vterm, opt->jo_ansi_colors);
  else
    init_vterm_ansi_colors(term->t1_vterm);
  #endif

  /* This may change a string in "argvar". */
  term->t1_job = job_start(argvar, argv, opt, TRUE);
  if (term->t1_job != NULL)
terminal.c          c  98%  5882: 5
```

This is work in progress..., Not support on gVim

`:echo has("new-feature")`

Suggestion 1: Support DRCS Sixel on `:terminal`

Note that this is not fully DRCS Sixel support on Vim. Just on `:terminal`. I have idea to support DRCS Sixel on Vim but I don't have patch yet.

`:echo has("new-feature")`

Suggestion 1: Support DRCS Sixel on `:terminal`

If Vim will support Sixel:

- Graphical sign icon
- Nyancat on statusline

:echo has("new-feature")

Suggestion 1: Support DRCS Sixel on :terminal

```
let s:dir = expand('<file>:h:h') . '/data'
let s:index = 0
let s:images = []

function! s:nyan(...)
  call writefile([printf("\x1b[s\x1b[%d;%dH", &lines-&cmdheight, &columns-7)],
  call writefile(s:images[s:index % len(s:images)], "/dev/tty", "b")
  call writefile(["\x1b[u"], "/dev/tty", "b")
  let s:index += 1
endfunction

function! s:start()
  for i in range(16)
    call add(s:images, readfile(printf('%s/nyan%03d.drcs', s:dir, i), 'b'))
  endfor
  call timer_start(50, function('<SID>nyan'), {'repeat': -1})
endfunction

call s:start()
~
~
nyan.vim
"nyan.vim" line 16 of 19 --84%-- col 21
```

```
:echo has("new-feature")
```

Suggestion 1: Support DRCS Sixel on :terminal



```
:echo has("new-feature")
```

Suggestion 2
Add `ch_listen()`.

:echo has("new-feature")

Suggestion 2: `ch_listen()` to listen socket

Current implementation of channel doesn't support listening socket. We often want to listen socket...

`:echo has("new-feature")`

Suggestion 2: `ch_listen()` to listen socket

Current implementation of channel doesn't support listening socket. We often want to listen socket...

Really?

:echo has("new-feature")

Suggestion 2: ch_listen() to listen socket

OBVIOUSLY!

Current implementation of channel doesn't support listening socket. We often want to listen ...

Really?



```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

Then, I added `ch_listen()` on Vim

```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

```
" listen port 8888  
let s:ch = ch_listen(  
  \ "127.0.0.1:8888",  
  \ {"callback": function("Accept")})
```



```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

```
function! Accept(ch, addr) abort
    " read/write on a:ch
    " addr is remote host:port
endfunction
```

```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

This make be possible do broadcasting for Vim.

`:echo has("new-feature")`

Suggestion 2: `ch_listen()` to listen socket

For example, it's possible to write web server.



mattn 15:51

vimconf2018 に来てください。本物のウェブアプリ(ただし Vim script)をお見せしますよ。

```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

But we know Vim script can't handle
bytes array contains NUL byte.

```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

It's not possible to serve binary file like image file
eventhough Vim might work as Web server.

OMG

```
:echo has("new-feature")
```

Suggestion 2: `ch_listen()` to listen socket

So, I added BLOB type on Vim

```
:echo has("new-feature")
```

Suggestion 3
Add BLOB type.

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" Current implementation:  
" readfile return array of strings  
" as separated strings with NUL.  
let arr = readfile("binary.dat", "b")  
" ["foo", "bar"]
```



```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" readfile with flag B return BLOB  
let b = readfile("binary.dat", "B")
```

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" writefile with flag B write binary file  
call writefile(b, "binary.dat")
```

`:echo has("new-feature")`

Suggestion 3: BLOB type

" BLOB is similar to List
echo b
[97,98,0,99,10]

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" get element  
echo b[1]  
98
```

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" iterable  
for c in b  
  echo c  
endfor
```

`:echo has("new-feature")`

Suggestion 3: BLOB type

```
" new literal 0zXXXXXXXXXX  
echo 0zFF0D352F  
[255,13,53,47]
```

`:echo has("new-feature")`

Suggestion 3: BLOB type

`" concat BLOB`

`echo 0xFF00 + 0x00FF`

`[255,0,0,255]`

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" append to BLOB  
call add(b, 1)
```



```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" remove element from BLOB  
call remove(b, 2)
```

`:echo has("new-feature")`

Suggestion 3: BLOB type

" index by byte number
echo index(b, 0x00)

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

" filter with condition

```
echo filter(b, {x-> x>0x50})
```

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" map with new value  
echo map(b, {x-> x+1})
```

`:echo has("new-feature")`

Suggestion 3: BLOB type

" send BLOB to channel
call `ch_sendraw(ch, b)`

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

```
" read BLOB from channel  
let b = ch_readblob(ch)
```

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

" Not supported yet
call libcall("libmyhack.so", "evil", b)

`:echo has("new-feature")`

Suggestion 3: BLOB type

Now this is possible to send NUL byte into channel.

:echo has("new-feature")

Suggestion 3: BLOB type

demo

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

BLOB is useful to handle bytes not just for
web server. Maybe ...

```
:echo has("new-feature")
```

Suggestion 3: BLOB type

I'll send this patch BLOB/ch_listen as
GitHub pull request in later.

:rewind



Of course, I really understand that Vim is text editor and not Emacs. Those features are not required for editing text. However, as that most people didn't believe that Vim support terminal feature, we don't know what feature coming. So those features may possibly be included.

:w summary

Summary...

:w summary

If you can write code, please send your patches to vim-dev. If you can't write English well, send it to vim-jp. We will review your code and introduce your patch to vim-dev.

:w summary

We will assist your dreams come true.

:q!

Thanks

May the Vim be with you.

:q!

Question?