

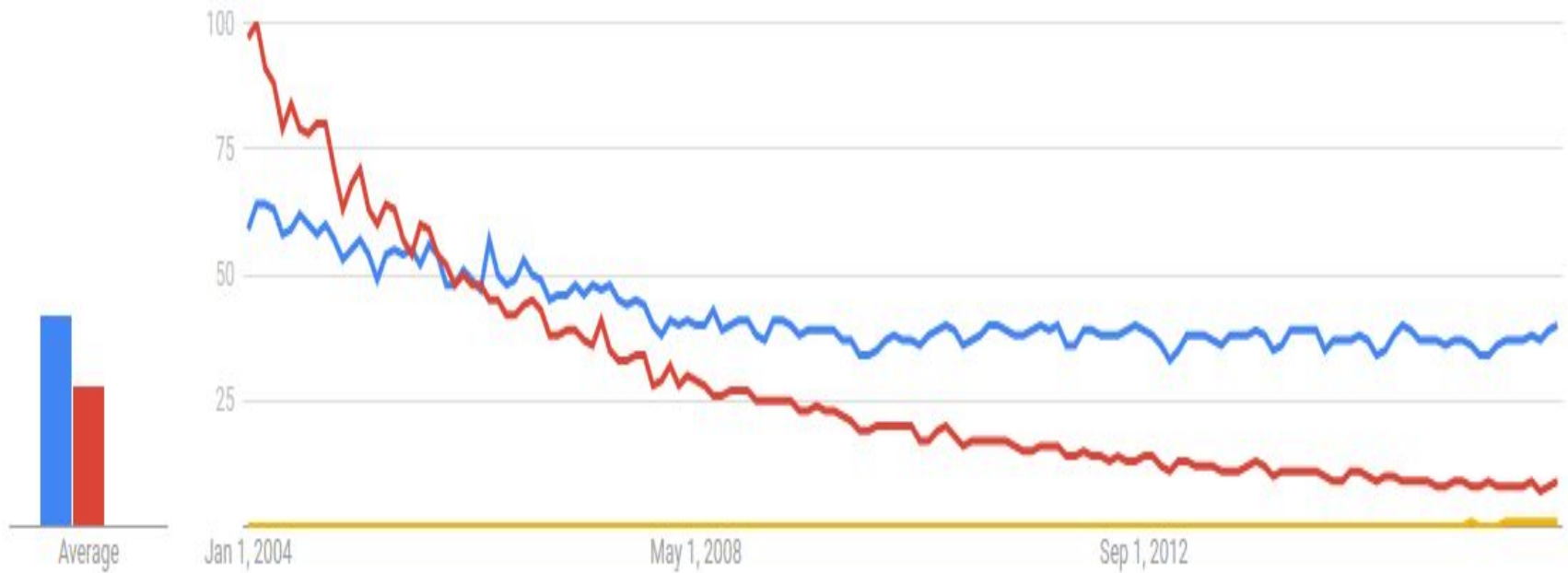


Recent Developments

Bram Moolenaar
SFI Krakow April 2018

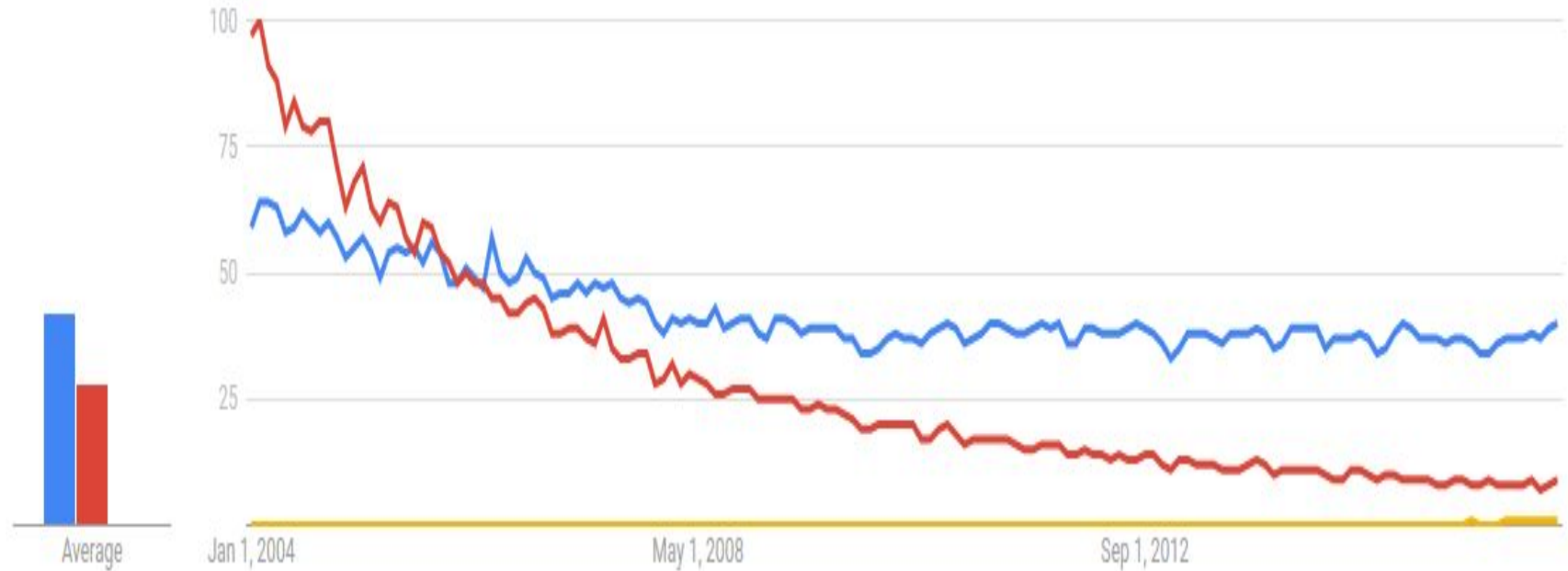
Vim popularity

Interest over time ?



Vim popularity

Interest over time ?

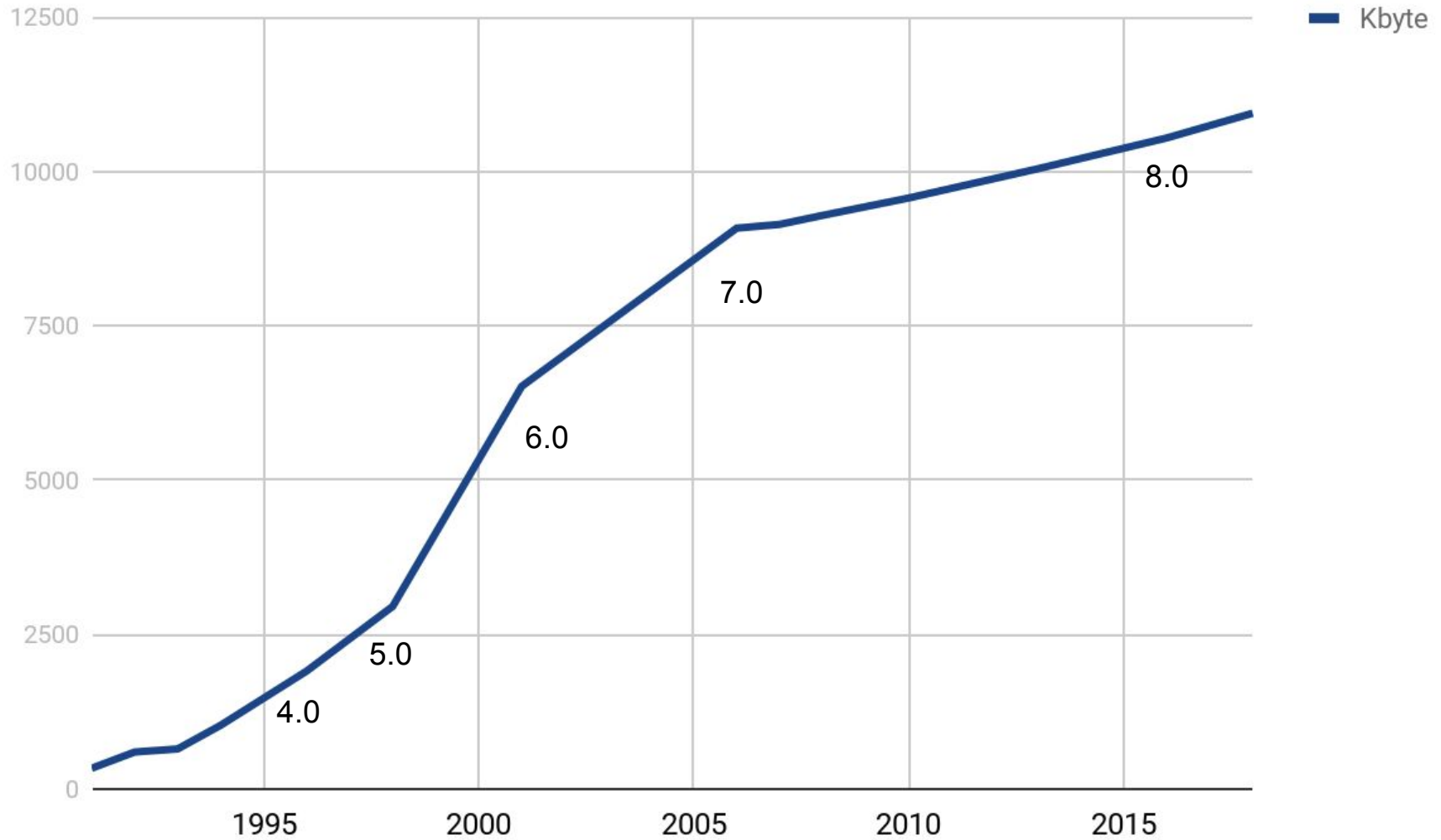


● Vim
Text editor

● Emacs
Software

● Neovim
Search term

Vim code size



src/*.[ch]

Impactful Changes

One has only limited time available to write code.
What to work on next to be most impactful?



Impactful Changes

One has only limited time available to write code.
What to work on next to be most impactful?

1. Work on a feature for 2-3 months tops

Impactful Changes

One has only limited time available to write code.
What to work on next to be most impactful?

1. Work on a feature for 2-3 months tops
2. Build one feature on top of another one

Impactful Changes

One has only limited time available to write code.
What to work on next to be most impactful?

1. Work on a feature for 2-3 months tops
2. Build one feature on top of another one
3. Understand the user

Impactful Changes

One has only limited time available to write code.
What to work on next to be most impactful?

1. Work on a feature for 2-3 months tops
2. Build one feature on top of another one
3. Understand the user

So, how did this work for Vim?

Vim first releases

1991: Vim 1.14 distributed on Fish disk

Impactful changes in Vim 1.27:

- Port to Unix
- Port to MS-DOS

More supported systems == more users



Vim 2.0

Impactful changes:

- `:make` and error parsing

Please users: Efficient edit - build - fix cycles

Vim 2.0

Impactful changes:

- `:make` and error parsing

Please users: Efficient edit - build - fix cycles

- Vi compatibility

Please distributors (Vim on every Linux and Mac system)



Vim 3.0

Impactful changes:

- Multiple windows and buffers

Make use of larger screens and more memory

Vim 3.0

Impactful changes:

- Multiple windows and buffers

Make use of larger screens and more memory

- Swap file

Reliability, user trust



Vim 4.0

Impactful changes:

- Help in a window

More complexity requires more support

Vim 4.0

Impactful changes:

- Help in a window

More complexity requires more support

- Autocommands

Extensibility, you can't build everything yourself

Vim 4.0

Impactful changes:

- Help in a window

More complexity requires more support

- Autocommands

Extensibility, you can't build everything yourself

- MS-Windows port

Reality: most users are there



Vim 5.0

Impactful changes:

- Syntax highlighting

Making use of faster computers and better screens

Vim 5.0

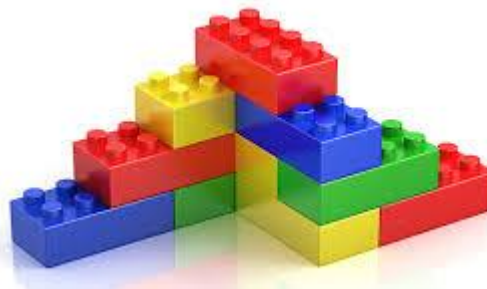
Impactful changes:

- Syntax highlighting

Making use of faster computers and better screens

- Vim script

Extensibility, you can't build everything yourself



Vim 6.0

Impactful changes:

- Unicode support
ASCII is no longer the standard

Vim 6.0

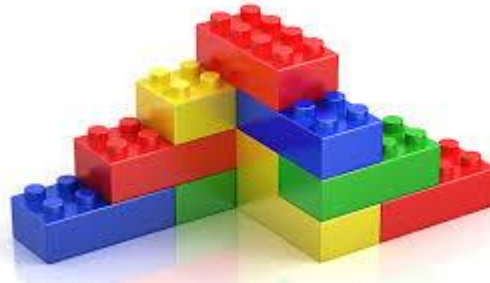
Impactful changes:

- Unicode support
ASCII is no longer the standard
- Automatic indenting
Users are lazy

Vim 6.0

Impactful changes:

- Unicode support
ASCII is no longer the standard
- Automatic indenting
Users are lazy
- Plugins
Extensibility, you can't build everything yourself

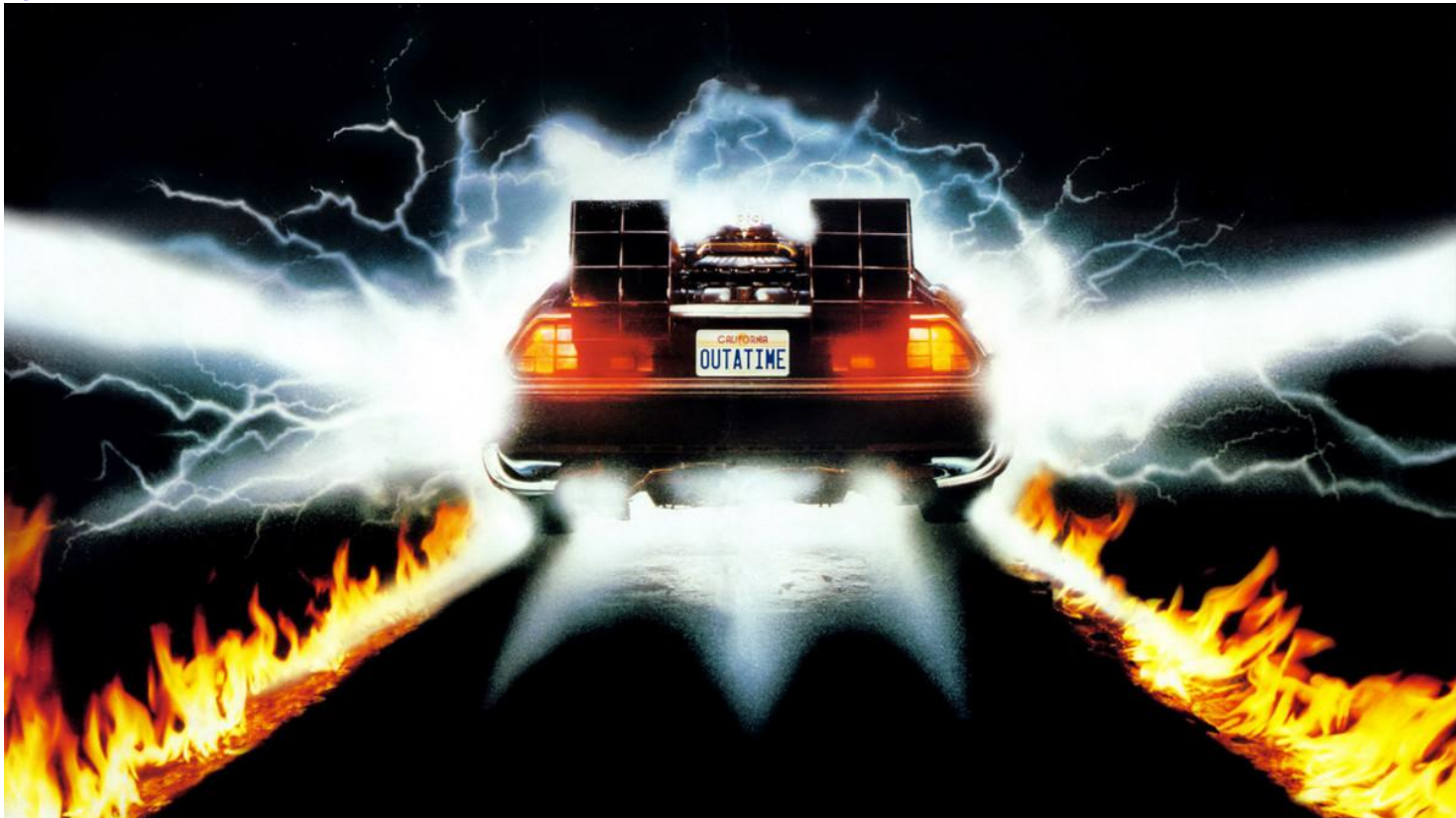


Vim 7.x

Impactful changes:

- Persistent undo

Don't worry, you can go back in time (and back to the future)

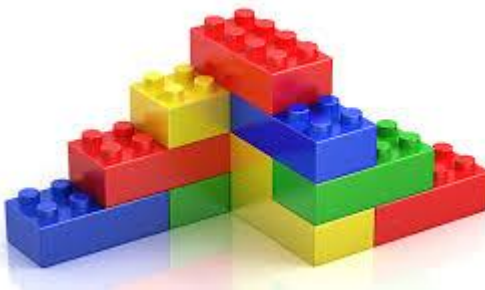


Vim 8.0

Impactful changes:

- Jobs, Channels and Timers

Extensibility, you can't build everything yourself
Make use of more powerful computers



Vim 8.1

Impactful changes?



Vim 8.1

Impactful changes:

- Terminal window

Why?



Vim 8.1

Impactful changes:

- Terminal window

Why?

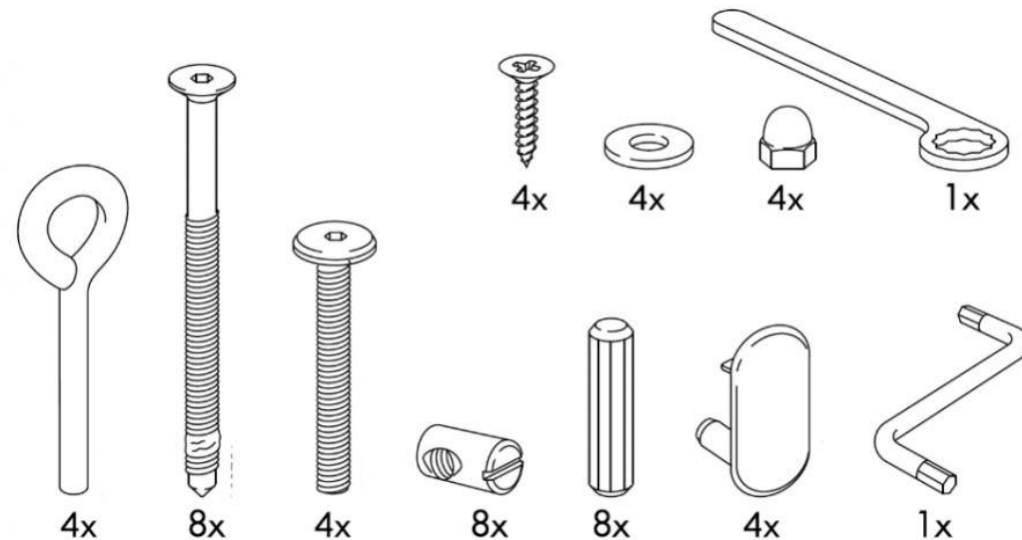
I want to debug Vim over ssh, which requires:

1. Window running gdb
2. Window running program being debugged
3. Window(s) to edit source code

Terminal debugger

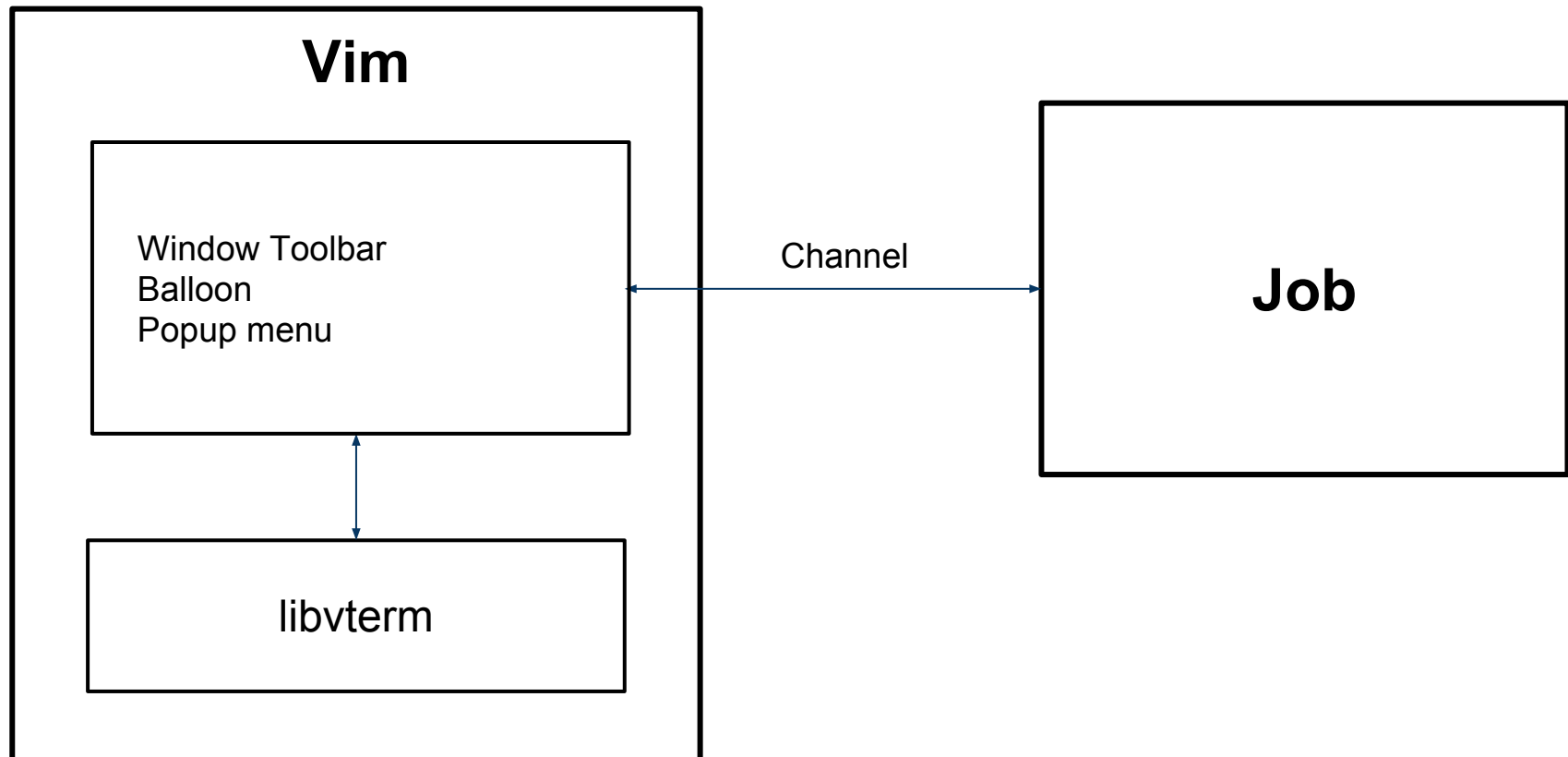
Parts needed:

1. Terminal emulator: libvterm
2. Jobs and channels
3. Window toolbar for Step/Next/Continue...
4. Balloon to show variable values
5. Popup menu

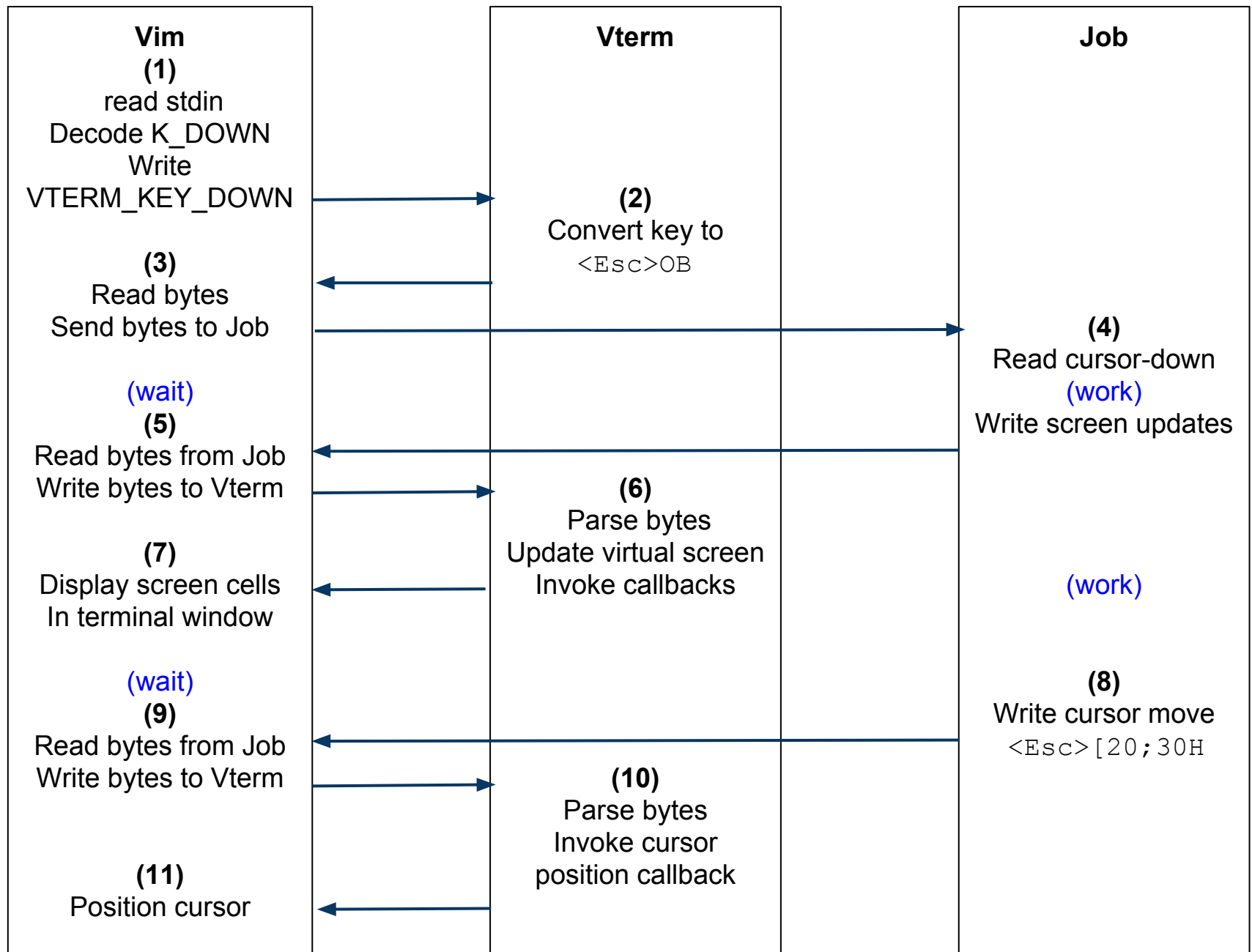


Terminal window

How the parts are put together:



Terminal window



Terminal debugger

Debugger demo

Terminal debugger

The image shows a terminal window with a debugger interface. The left pane contains the following text:

```
new-ui mi /dev/pts/16  
Reading symbols from vim...done.  
(gdb) new-ui mi /dev/pts/16  
New UI allocated  
(gdb) █
```

The right pane has a toolbar with buttons: Step, Next, Finish, Cont, Stop, Eval.

At the bottom of the terminal, there are two status bars:

- A green bar: `!gdb [running] 0,0-1 All`
- A black bar: `gdb program [active] 0,0-1 All [No Name] 0,0-1 All`

Terminal debugger

```
new-ui mi /dev/pts/17
Reading symbols from vim...done.
(gdb) new-ui mi /dev/pts/17
New UI allocated
(gdb)

!gdb [running] 0,0-1 All

gdb program [active] 0,0-1 All ~/vim/vim80/src/ex_cmds.c 6205,1 73%
```

```
Step Next Finish Cont Stop Eval
return FALSE;
}
#endif
/*
 * ":help": open a read-only window on a help file
 */
void
ex_help(exarg_T *eap)
{
    char_u      *arg;
    char_u      *tag;
    FILE        *helpfd;      /* file descriptor of help file */
    int         n;
    int         i;
    win_T       *wp;
    _matches;
    atches;
    ty_fnum = 0;
    _fnum = 0;
    f;
    # Select Paragraph ANG
    Select Line ;
    Select Block ng;
    # Select All
    #
    Set breakpoint _KeyTyped = KeyTyped;
    # Clear breakpoint
    Evaluate
    if (eap != NULL)
    {
        /*
         * A ":help" command ends at the first LF, or at a '|' that is
         * followed by some text. Set nextcmd to the following command.
         */
        for (arg = eap->arg; *arg; ++arg)
    }
```

Terminal debugger

```
new-ui mi /dev/pts/17
Reading symbols from vim...done.
(gdb) new-ui mi /dev/pts/17
New UI allocated
(gdb) run
Starting program: /home/mool/vim/vim80/src/vim
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".

Breakpoint 1, ex_help (eap=0x7fffffff4f0) at ex_cmds.c:6215
6215     if (eap != NULL)
(gdb)

!gdb [running] 0,0-1 All

VIM - Vi IMproved

      version 8.0.1615
      by Bram Moolenaar et al.
Vim is open source and freely distributable

  Become a registered Vim user!
type  :help register<Enter>  for information

type  :q<Enter>              to exit
type  :help<Enter> or <F1>   for on-line help
type  :help version8<Enter> for version info

[he gui

gdb program [[No Name] - VIM] 0,0-1 All ~/vim/vim80/src/ex_cmds.c 6215,5 73%
:Break
```

```
Step  Next  Finish  Cont  Stop  Eval
return FALSE;
}
#endif
/*
 * ":help": open a read-only window on a help file
 */
void
ex_help(exarg_T *eap)
{
    char_u      *arg;
    char_u      *tag;
    FILE        *helpfd;      /* file descriptor of help file */
    int         n;
    int         i;
    win_T       *wp;
    int         num_matches;
    char_u      **matches;
    char_u      *p;
    int         empty_fnum = 0;
    int         alt_fnum = 0;
    buf_T       *buf;
#ifdef FEAT_MULTI_LANG
    int         len;
    char_u      *lang;
#endif
#ifdef FEAT_FOLDING
    int         old_KeyTyped = KeyTyped;
#endif
>> if (eap != NULL)
    {
        /*
         * A ":help" command ends at the first LF, or at a '|' that is
         * followed by some text.  Set nextcmd to the following command.
         */
        for (arg = eap->arg; *arg; ++arg)
        {
```

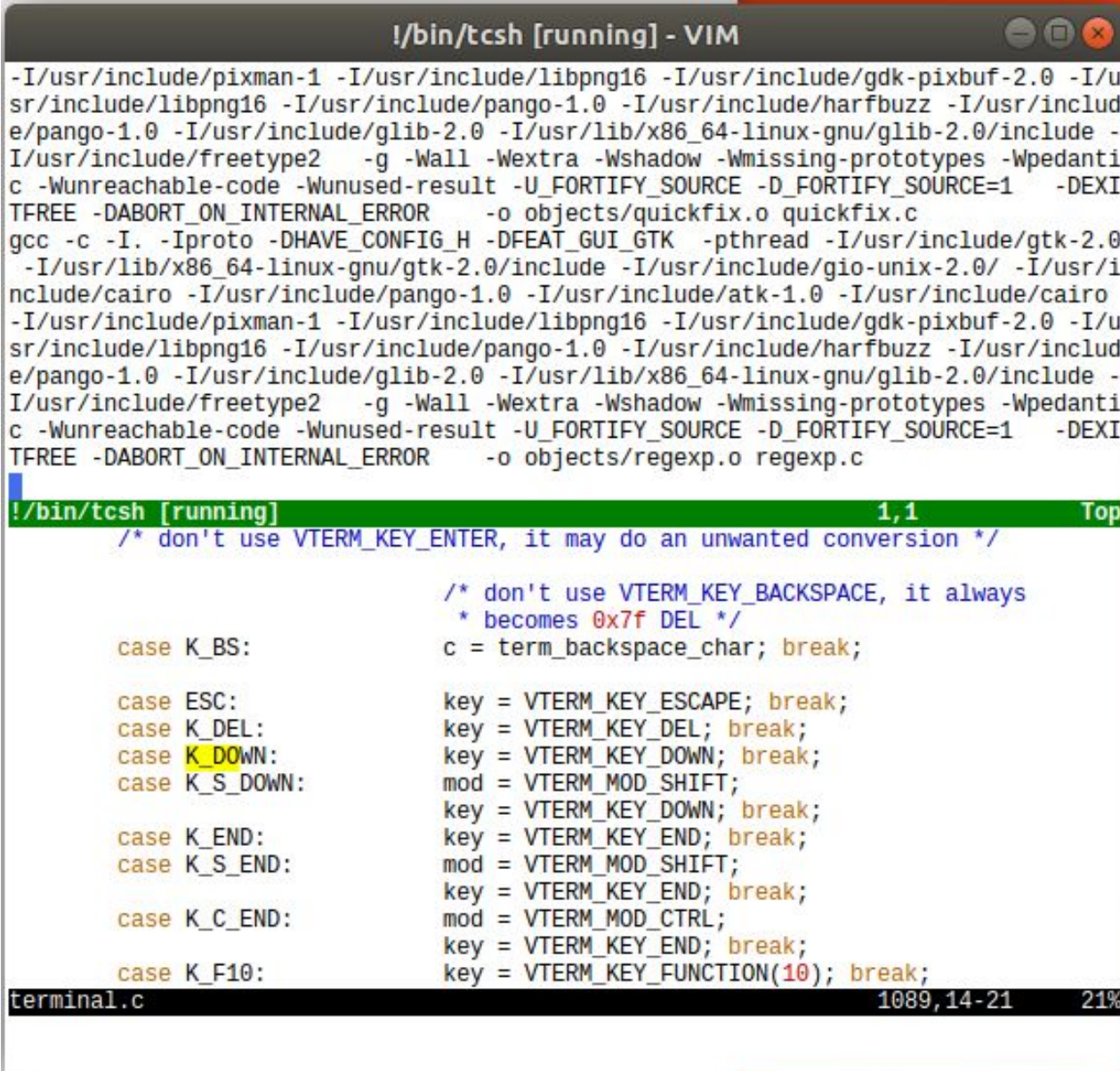

Terminal window

What else can you use it for?



Terminal window

Run make in a terminal window



```
!/bin/tcsh [running] - VIM
-I/usr/include/pixman-1 -I/usr/include/libpng16 -I/usr/include/gdk-pixbuf-2.0 -I/
sr/include/libpng16 -I/usr/include/pango-1.0 -I/usr/include/harfbuzz -I/usr/includ
e/pango-1.0 -I/usr/include/glib-2.0 -I/usr/lib/x86_64-linux-gnu/glib-2.0/include -
I/usr/include/freetype2 -g -Wall -Wextra -Wshadow -Wmissing-prototypes -Wpedanti
c -Wunreachable-code -Wunused-result -U_FORTIFY_SOURCE -D_FORTIFY_SOURCE=1 -DEXI
TFREE -DABORT_ON_INTERNAL_ERROR -o objects/quickfix.o quickfix.c
gcc -c -I. -Iproto -DHAVE_CONFIG_H -DPEAT_GUI_GTK -pthread -I/usr/include/gtk-2.0
-I/usr/lib/x86_64-linux-gnu/gtk-2.0/include -I/usr/include/gio-unix-2.0/ -I/usr/i
nclude/cairo -I/usr/include/pango-1.0 -I/usr/include/atk-1.0 -I/usr/include/cairo
-I/usr/include/pixman-1 -I/usr/include/libpng16 -I/usr/include/gdk-pixbuf-2.0 -I/u
sr/include/libpng16 -I/usr/include/pango-1.0 -I/usr/include/harfbuzz -I/usr/includ
e/pango-1.0 -I/usr/include/glib-2.0 -I/usr/lib/x86_64-linux-gnu/glib-2.0/include -
I/usr/include/freetype2 -g -Wall -Wextra -Wshadow -Wmissing-prototypes -Wpedanti
c -Wunreachable-code -Wunused-result -U_FORTIFY_SOURCE -D_FORTIFY_SOURCE=1 -DEXI
TFREE -DABORT_ON_INTERNAL_ERROR -o objects/regexp.o regexp.c

!/bin/tcsh [running] 1,1 Top
/* don't use VTERM_KEY_ENTER, it may do an unwanted conversion */
/* don't use VTERM_KEY_BACKSPACE, it always
 * becomes 0x7f DEL */
case K_BS: c = term_backspace_char; break;
case ESC: key = VTERM_KEY_ESCAPE; break;
case K_DEL: key = VTERM_KEY_DEL; break;
case K_DOWN: key = VTERM_KEY_DOWN; break;
case K_S_DOWN: mod = VTERM_MOD_SHIFT;
key = VTERM_KEY_DOWN; break;
case K_END: key = VTERM_KEY_END; break;
case K_S_END: mod = VTERM_MOD_SHIFT;
key = VTERM_KEY_END; break;
case K_C_END: mod = VTERM_MOD_CTRL;
key = VTERM_KEY_END; break;
case K_F10: key = VTERM_KEY_FUNCTION(10); break;
terminal.c 1089, 14-21 21%
```


Testing old style

.in file:

```
STARTTEST
:so small.vim
:set belloff=all
/Start cursor here
vaBiBD:?Bug?,/Piece/-2w! test.out
/^- Bug
:s/u/~u~/
:s/i/~u~/
:s/o/~~/
:..w >>test.out
```

.ok file:

- Bug in "vPPPP" on this text (Webb):
 {
 }
- Bug uuun "vPPPP" uuuuuuuuun this text (Webb):

Testing new style

.vim file:

```
func Test_move_cursor()
  new
  call setline(1, ['aaa', 'bbb', 'ccc', 'ddd'])

  call cursor([1, 1, 0, 1])
  call assert_equal([1, 1, 0, 1], getcurpos()[1:])
  call cursor([4, 3, 0, 3])
  call assert_equal([4, 3, 0, 3], getcurpos()[1:])

  call cursor(2, 2)
  call assert_equal([2, 2, 0, 2], getcurpos()[1:])
  " line number zero keeps the line number
  call cursor(0, 1)
  call assert_equal([2, 1, 0, 1], getcurpos()[1:])
```

Testing with a screenshot

.vim file:

```
func Test_popup_position()
  if !CanRunVimInTerminal()
    return
  endif
  call writefile([
    \ '123456789_123456789_123456789_a',
    \ '123456789_123456789_123456789_b',
    \ '          123',
    \ ], 'Xtest')
  let buf = RunVimInTerminal('Xtest', {})
  call term_sendkeys(buf, ":vsplit<CR>")

  " default pumwidth in left window: overlap in right window
  call term_sendkeys(buf, "GA<C-N>")
  call VerifyScreenDump(buf, 'Test_popup_position_01', {'rows': 8})
```

Terminal window

Testing with a screenshot diff

```
dump diff [finished] - GVIM
File Edit Tools Syntax Buffers Window Plugin Help
123456789_123456789_123456789_a | 123456789_123456789_123456789_a
123456789_123456789_123456789_b | 123456789_123456789_123456789_b
      123456789_123456789_12345 | 123456789_123456789_12345
6789_b | 6789_b
~ | 123456789_123456789_123456789_a
~ | 123456789_123456789_123456789_b
~ |
~ |
-----
X | X
  | bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
  | bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb
  |
-----
123456789_123456789_123456789_a | 123456789_123456789_123456789_a
123456789_123456789_123456789_b | 123456789_123456789_123456789_b
      123456789_123456789_12345 | 123456789_123456789_12345
6789_a | 6789_a
~ | 123456789_123456789_123456789_a
~ | 123456789_123456789_123456789_b
~ |
~ |
~ |
-----
dump diff [finished] 4,7 ALL
[No Name] 0,0-1 ALL
```

Vim 8.1

Release: “in a few weeks”



The end

Questions?