

Vim Internals

A Gentle Introduction

Speaker: Jason Franklin

Date: 2019-10-23

Slides: <https://bit.ly/31DnxdT>

Files: <https://bit.ly/2PcgFle>

The Roadmap



- Why Vim?
- Going to the source.
- Fixing a “high-level” bug! (VimL)
- Fixing a “low-level” bug! (C)
- Where to?

Part I: Why Vim?

Greetings...



```
#include <stdio.h>
```

```
int main(void) {
```

```
    printf("Hello, World!\n");  
}
```

Greetings...



```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello, World!");  
    }  
}
```

Greetings...



```
#include <stdio.h>
```

```
int main(void) {
```

```
    printf("Hello, World!\n");  
}
```

Greetings...



```
ssize_t write(int fd, const void *buf, size_t count);
```

It's turtles all the way down.



Why Vim?



- Vim is forever. Well, at least **since 1976**.

<https://www.youtube.com/watch?v=VADudzQGvU8> (1:08:44)

- Vim allows you to **achieve mastery**.

<https://www.youtube.com/watch?v=1qLstQV2j8w> (6:40)

- Vim demands that you **grok the toolchain**.

- Vim is **free!**

Part II: Going to the source.

You will need:



- The *repository*.

```
$ git clone https://github.com/vim/vim.git
```

- Some *packages*.

```
$ sudo apt install gcc libncurses-dev \  
python3-dev xorg-dev
```

- Or... some packages the *easy way*!

```
$ sudo apt build-dep vim
```

You will need to run:



```
$ cd vim/src/
```

```
$ make
```

```
$ ./vim --clean; # We did it!
```

```
$ make test
```

Part III: A high-level bug.

The process.



- 1)Identify the bug with a precise reproduction.
- 2)Confirm the bug still exists.
- 3)Research: Is someone else working on it?
- 4)Fix the bug if you can!
- 5)Report the bug *with the solution*. (fastest!)

<?php

\$b = -0b0111010;

\$b = -0B0111010;

\$o = -011127;

\$o = -019997;

\$n = -0x1382471FFF;

\$n = -0X1382471FFFA;

\$d = -10;

~

~

~

~

~

~

~

~

~

~

~

11

2

2

2

2

2

2

2

2

2

2

2

2

From the PHP website:



```
decimal      : [1-9][0-9]*  
              | 0  
  
hexadecimal  : 0[xX][0-9a-fA-F]+  
  
octal        : 0[0-7]+  
  
binary       : 0[bB][01]+  
  
integer      : decimal  
              | hexadecimal  
              | octal  
              | binary
```

The new rules for Numbers:



```
" In the file "vim/runtime/syntax/php.vim" ...  
syn match phpNumber "\<\%([1-9]\d*\|0\)\>" contained display  
syn match phpNumber "\<0[xX]\x\+\>" contained display  
syn match phpNumber "\<0[o]\+\>" contained display  
syn match phpNumber "\<0[bB][01]\+\>" contained display
```


\$

\$

\$

\$

\$

\$

\$

\$

2

2

2

2

2

2

2

2

2

2

Summary for bug #1



- How did I know what to do?
- What did I do next?
- How did my discussion with the maintainer proceed?
- See the patch: **php.diff**

Demo.

Part IV: A low-level bug.

The details...



- The patch: <https://bit.ly/2Myr0eg>
- The email: <https://bit.ly/2o1KERA>
- Now for a demo and some analysis!

Demo.

The test (part I)



```
func Test_insert_cleared_on_switch_to_term()
    CheckFeature terminal

    set showmode
    terminal
    wincmd p

    call feedkeys("i\<C-O>", 'ntx')
    redraw

    " The "-- (insert) --" indicator should be visible.
    let chars = map(range(1, &columns), 'nr2char(screenchar(&lines, v:val))')
    let str = trim(join(chars, ''))
    call assert_equal('-- (insert) --', str)
```

The test (part II)



```
call feedkeys("\<C-W>p", 'ntx')
redraw

" The "-- (insert) --" indicator should have been cleared.
let chars = map(range(1, &columns), 'nr2char(screenchar(&lines, v:val))')
let str = trim(join(chars, ''))
call assert_equal('', str)

set showmode&
%bw!
endfunc
```

The fix!



```
#ifdef FEAT_TERMINAL
    if (bt_terminal(curwin->w_buffer))
        // terminal is likely in another mode
        redraw_mode = TRUE;
#endif
```

The perfect patch...



- 1) ... fixes a bug.
- 2) ... adds a test case.
- 3) ... reduces complexity.

This is the “hat trick” in software development.

Part V: Where to?

The future of Vim...



- 1) The Matrix
- 2) Star Trek
- 3) Blade Runner
- 4) Mad Max
- 5) The Road

A Quote.

The end.



Questions?