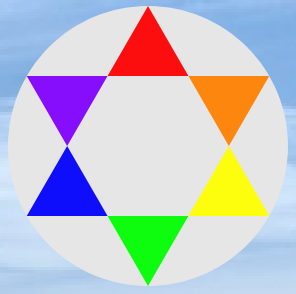
and Compatibility

妹尾賢
SENOO, Ken
✉ contact@senooken.jp
https://social.senooken.jp/senooken

2017-11-04

VimConf 2017

URL: https://senooken.jp/public/20171104/



Trends for Development Style

There Are 2 Trends for Development Style

1. Updating（最新化）
2. Keeping（維持）

Updating（最新化）

Using latest technology (tools, API, library, device)

↓

Achieving high **performance** & **function**

Example

- CUDA (GPGPU)
- Android/iOS

Keeping（維持）

Using de jure standard or common technology

(ISO, IEEE, W3C, standard library, etc.)

↓

Achieving high **compatibility** & **durability**

Example

- XML, HTML, JSON
- HTTP

Updating vs. Keeping

Updating

- High performance
- High function

Keeping

- High compatibility
- High durability

Also Vim has same trends.

Updating

- New plugin
- New feature/interface
- New function/option

Keeping

- Supporting OS
- Supporting language
- Bug fix & improvement

Importance of Keeping

- No keeping, no working.
- No using, no meaning.
- ***Updating is based on keeping.***

cybozu Live

無料グループウェアサイボウズLiveは、２０１９年４月１５日をもってサービスを終了させていただきます。

サイボウズLiveは、企業外で使う「セカンドグループウェア」というコンセプトの元、2010年10月に正式提供を開始いたしました。

システムの老朽化などにより、今後も安定的にサービスを継続するには抜本的な作り直しなどの投資が必要になっていること、有料サービスへのさらなる投資が必要になっていることを背景に、「サイボウズLive」終了を決断いたしました。

サービス終了に伴い、ご利用中のお客様には、多大なご迷惑をおかけいたしますことを深くお詫び申し上げます。

Example: End of Life for cybozu Live

- Groupware web service "cybozu Live" by Cybozu, Inc in Japan.
- Start from 2009-11-26.
- ***2 million users.***
- Discontinuing was announced on 2017-10-24.
- End of support on 2019-04-15.

- Cybozu have used ***Java Seasar projects*** in past.
- Many Seasar projects are ***End of Life*** (EOL) now.
- This announcement is due to Seasar EOL probably.

> - *Ref: サイボウズLive - 西尾泰和のScrapbox - Scrapbox*
> - *Ref: サイボウズLiveサービス終了のお知らせ | 無料グループウェア サイボウズLive*

サービス終了に伴い、ご利用中のお客様には、多大なご迷惑をおかけいたしますことを深くお詫び申し上げます。

How do we keep our development?

Methodology for Keeping

POSIXism (POISX fundamentalism)

POSIX原理主義


Write Once,
Work
Anytime/Anywhere

POSIX and POSIXism

POSIX（Portable Operating System Interface）

移植可能なオペレーティングシステムのインターフェイス

- Called IEEE 1003 family (1003.n) in official.
- Define portable OS API (mainly C API and shell, commands) like UNIX.
- Since 1988, it is available for **more than 25 years**.
- **Linux, BSD, Mac** is almost conforming to POSIX.
- Developing for conforming to POSIX, work **anytime/anywhere** on POSIX conformal OS.

POSIXism (POSIX fundamentalism：POSIX原理主義)

- Development methodology for **keeping portability** (exchangeability) like POSIX.
- Keeping portability, decreasing dependency for specific product.
- If there is ***vulnerability or discontinuing***, we can keep by **switching alternative** product.

Good/Bad Point of POSIXism

Good

- Maintenance free
- Supporting from old OS to latest OS
- Easy for installing

Bad

- Required for tips for high compatibility and durability
- Applying to ***edge of technology*** is difficult for portability
- There is ***challenging field*** like binary data processing.

3 Guidelines for POSIXism

Keeping Portability（移植可能性担保）

Developing with working ***2 implementation at least***.
複数のコマンド・実装での動作を確保できるように開発

Conforming to POSIX（POSIX準拠）

Developing with conforming to POSIX (***Shell script and C99***)
POSIX規格に準拠したシェルスクリプト・C99によるプログラミング

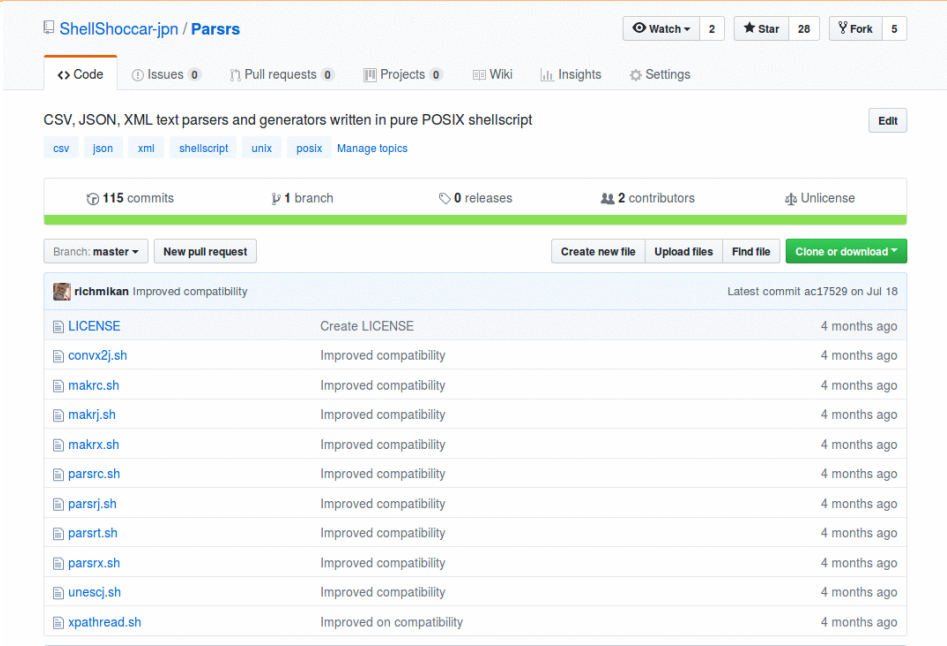
Conforming to W3C（W3C準拠）

Developing with conforming to W3C (***HTML+CSS+JavaScript***) for web app
Webアプリにおけるクライアントサイドの開発ではW3C勧告に準拠

- **Essence of POSIXism is to keep portability**.
- We can use de jure standard (IEEE, ISO, ECMA, JIS) or official reference.

Application for POSIXism

Shell script with conforming to POSIX
Parsrs
https://github.com/ShellShoccar-jpn/Parsrs



[ShellShoccar-jpn/Parsrs: CSV, JSON, XML text parsers and generators written in pure POSIX shellscript]

- General parser and generator for JSON, XML.
- **Good bye, jq, jo, xmllint**

Web app with conforming to W3C
Metropiper
http://lab-sakura.richlab.org/METROPIPER/HTML/MAIN.HTML

["Metropiper" for train aproaching information web app. (接近情報表示プログラム「メトロパイパー」)]

- Showing Tokyo metro train position info.
- Developed using parsrs.

Publication

Paper


デジタルプラクティス　DP目次へ戻る

デジタルプラクティス　Vol.8 No.4 (Oct. 2017)

推薦投稿論文

ソフトウェアの高い互換性と長い持続性を目指す POSIX中心主義プログラミング

松浦 智之[1]　大野 浩之[2]　當仲 寛哲[1]

[1]（有）ユニバーサル・シェル・プログラミング研究所　[2]金沢大学

ソフトウェアは，より高度な要求，あるいは時代とともに変化する要求に応えるべく，絶え間なくバージョンアップを繰り返している．しかし，多くのソフトウェアは，今自分の置かれた環境において求められる性能や機能を満たすことばかり偏重し，ほかの環境や将来の環境における互換性をあまり考慮していない．そこで筆者らは，UNIX系OSが最低限満たすべきとした仕様をまとめた国際規格であるPOSIX（Portable Operating System Interface）に着目した．POSIXは現状で多くのUNIX系OSが準拠している上に，1988年の初出以来，その仕様はほとんど維持されている．このような性質を持つ規格に極力準拠しながらプログラミングすることで，ソフトウェアは高い互換性と長い持続性を得られる可能性がある．そして，筆者らはこのようにしてPOSIXの仕様に極力準拠しながらプログラミングをする指針を具体的にまとめ，POSIX中心主義と名付けた．本稿では，POSIX中心主義としてまとめたプログラミング指針を提案するとともに，現在行っている互換性と長期持続性の検証について報告する．

Peer reviewed paper on Information Processing Society of Japan 2017-10-15
https://www.ipsj.or.jp/dp/contents/publication/32/S0804-R1601.html

HP


posixism.org

About　News　Repository　Publication　FAQ　Contact

Write Once,
Work Anytime/Anywhere

About

posixism.orgはPOSIX原理主義に関する情報を集約することを目的としたサイトです。

POSIXism

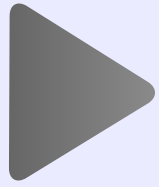
POSIX原理主義とは，高い互換性と長い持続性を目指したソフトウェアの開発技法です。

POSIX原理主義の主な利点と欠点を以下に掲載します。

posixism.org
http://posixism.org/

- POSIXism is published some media.
- Paper and HP.

Example for Applying POSIXism to Our Life



- Twitter is popular in Japan. But Twitter is only one, not portable.
- **If Twitter bans your account, it is the end.**
- Try ***GNU Social*** or ***Mastodon*** instead.

Vim and POSIXism

- POSIX is including ed, ex, vi for Vim family.
- Vim is used for a long time and many places.
- Vim ♡ POSIX

Applying POSIXism to Vim

Write .vimrc Once, Work Anytime/Anywhere

Vim Overview

Vim Inheritance

ed

↓

ex

↓

vi

↓

vim

Developers

Initial release	Name	Creator (Nick name)	Portrait	Bio
1969-08	ed	Kenneth Lane Thompson (Ken Thompson)	 (Wikimedia Commons)	• Born 1943-02-04 • American • Computer scientist • UNIX developer • UTF-8 encoding developer • Go language co-designer.
1976-09 1979-05	ex vi	William Nelson Joy (Bill Joy)	 (Wikimedia Commons)	• Born 1954-11-08 • American • Computer scientist • csh developer • Co-founder of Sun Microsystems
1988	vim	Bram Moolenaar	  (Wikimedia Commons) (http://www.zimbu.org/)	• Born 1961 • Germans • Programmer • Developer for Zimbu language.

Vi History

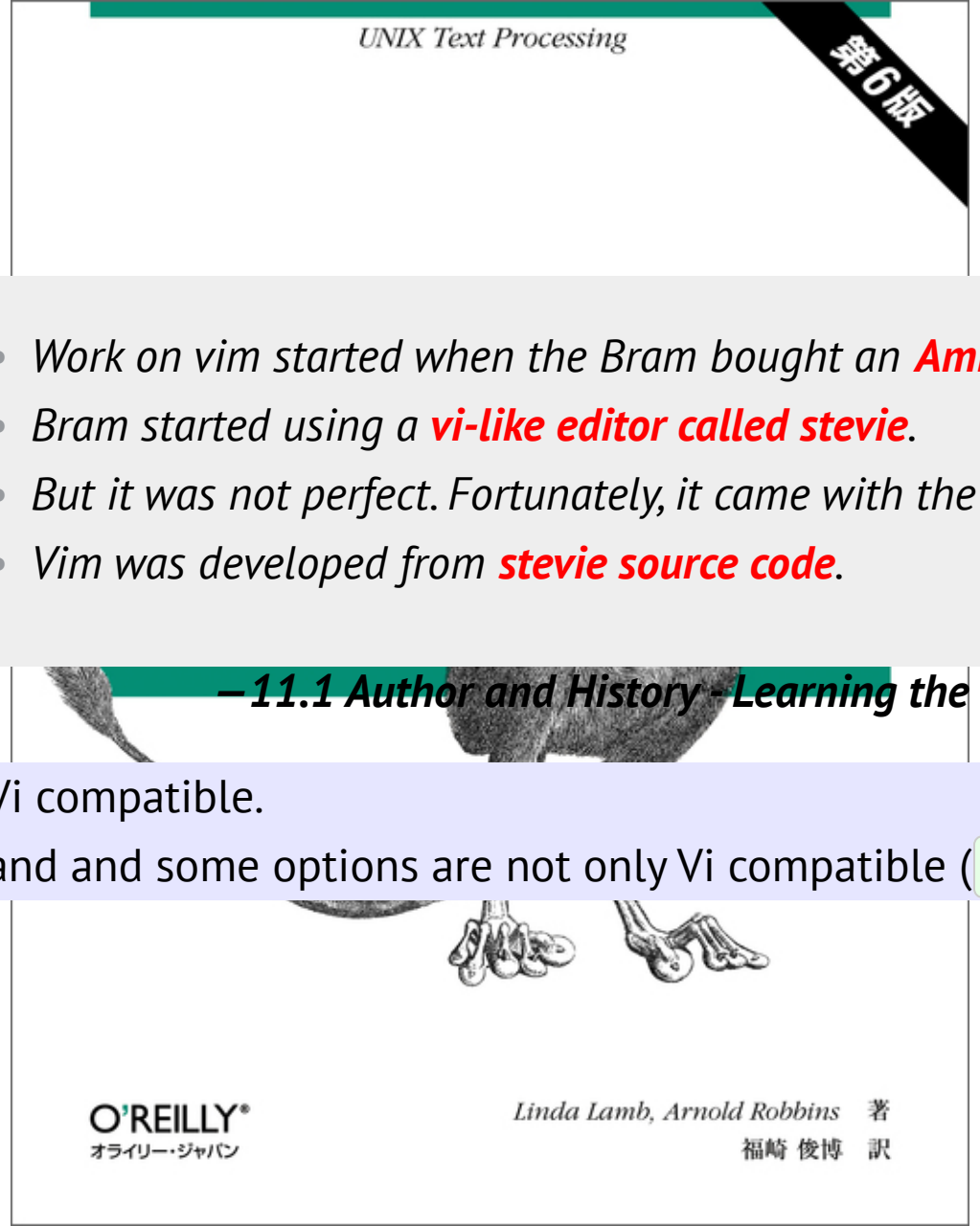
Vi = Visual Interface or Visual Editor

Date	Editor	Description
1969-08	ed	Initial release for line editor ed as a part of UNIX by Kenneth Lane Thompson.
1976-02	em	George Coulouris enhanced ed to make em (editor for mortals).
1976-09	ex 0.1	Bill Joy took code from em to make en, and then "extended" en to create ex 0.1.
1977-10	ex 1.1	adding a full-screen visual mode to ex.
1979-05	ex (vi) 2.0	vi as a hard link to ex.
1979-06-10	vi 2.7	last version for Joy as a leading developer
1979-11-01	vi 3.1	shipped with 3BSD
1980-08-20	vi 3.5	last version from Joy contribution
1981-10-16	vi 3.7	UNIX System V adopting vi. And distributed Solaris, HP-UX, Tru64 UNIX, AIX.
1987-06	STEVIE	STEVIE (ST Editor for VI Enthusiasts), a limited vi clone.
1988	Vim	Vi IMitation on the Amiga
1990-01	Elvis	Steve Kirkendall posted a new clone of vi, Elvis.
1994	nvi	Created nvi from Elvis 1.8 for 4.4 BSD Lite.

- Vi is super set of ex.
- After vi 3.7, many vi clones were born.
- vi and vi clone are available for UNIX like OS.

vi - Wikipedia

Birth of Vim



> - *Work on vim started when the Bram bought an **Amiga** computer.*
> - *Bram started using a **vi-like editor called stevie**.*
> - *But it was not perfect. Fortunately, it came with the source code.*
> - *Vim was developed from **stevie source code**.*
>
> ***—11.1 Author and History - Learning the vi Editor(入門vi 第6版)***

- Vim is almost Vi compatible.
- `:open` command and some options are not only Vi compatible (`:help vi-diff`).

Vim History

Date	Version	Description
1988	1.0	Vi IMitation on the Amiga (***same as first POSIX release year***)
1991-12-14	1.14	First public release
1992	1.22	Port to Unix and MS-DOS
1993-12-21	2.0	***Renamed to Vi IMproved***, horizontal scroll and 'wrap', wildcard expansion,
1994-08-16	3.0	Multiple buffers and windows
1996-05-21	4.0	GUI, autocmd, mouse, swap file, Windows NT & 95, OS/2, tag, viminfo
1998-02-19	5.0	Syntax highlighting, Vim script, perl and python, Win32 GUI, VMS, BeOS, Mac GUI,
2001-09-27	***6.0***	Folding, plugins, vertical split, diff, UTF-8, multi-language, netrw, quickfix
2002-03-24	6.1	Bug-fix release
2003-06-01	6.2	GTK2, arabic text, :try
2004-06-08	6.3	Bug-fix release
2005-10-15	6.4	Bug-fix release
2006-05-08	***7.0***	Vim script feature(list, dictionary, funcref, +=, -=, .=, vimball), spell checking, Omni completion, tab pages, vimgrep, location list
2007-05-12	7.1	Bug-fix release
2008-08-09	7.2	Floating points support.
2010-08-15	7.3	Persistent undo, encryption, conceal, lua, python3
2013-08-10	7.4	New regexp engine, better python interface
2016-09-12	***8.0***	Asynchronous I/O support, channels, jobs, timers, lambda and closure, packages etc.

Ref: Talk of TAKATA@VimConf 2016 *Introduction to Vim 8.0*

Which Vim Version Is It Possible?

- Vim has many feature with version up.
- Latest version is the best.
- But we are not available for latest always (offline, customer's, old VPS).

Example

- 8: Best. Full supporting latest features (asynchronous I/O, lambda, closure, package...)
- 7: Good. Many features enabled.
- 6: Bad. Many features disabled (List, dictionary, funcref, +=, -=, .=, Float, tab page).

Which version is it possible for .vimrc or plugin?

Concept for Default Vim Version

Time is going now. We need to fix the base date for default Vim version.

How to decide

- Latest POSIX release date: 2016-09-30
- The oldest vendor supporting OS.

Research the oldest Vim version
for the oldest vendor supporting OS
on latest POSIX (2016-09-30).

Real VPS Specification

OS	標準	CentOS 6.5 x86_64
	その他	ISOイメージファイルをアップロードしてお好みのOSをインストールできます。 ※以下のISOイメージは弊社にてあらかじめご用意しています。 CentOS 7.1 (64bit) CentOS 7.0-1406 (64bit) CentOS 5.9,5.10,5.11,6.2,6.3,6.4,6.5,6.6 (32bit/64bit) Debian 8.0,8.6 (32bit/64bit) Debian GNU/Linux 7 (32bit/64bit) Fedora 22 (32bit/64bit) FreeBSD 9.3,10.1,11.0 (32bit/64bit) Ubuntu 12.04LTS,14.04LTS,16.04 (32bit/64bit) Scientific Linux 7.1 (64bit) Scientific Linux 7 (64bit) Scientific Linux 6.2,6.3,6.4,6.5 (32bit/64bit) CoreOS Stable (64bit) Arch Linux (32bit/64bit)

Ref: スペック・機能一覧 | お名前.com VPS | 月額896円(税抜)からのVPS（KVM）

CentOS 5 looks like the oldest supported OS.

OS Default Vim Version

Type	OS	Release	End of life	vi/Vim version
	POSIX 2016	2016-09-30		
Linux	CentOS 5.0&5.11	2007-04-12	2017-03-31	**7.0**
Linux	Debian 7.0	2013-05-03	2018-05-31	7.3
Linux	Ubuntu 12.04	2012-04-26	2017-04-28	7.3
Linux	OpenSUSE	2013-11-19	2016-02-03	7.4
BSD	FreeBSD 9.0	2012-01-10	2016-12-31	None
UNIX	Mac OS X 10.6 (Snow Leopard)	2009-08-28	2014-02-25	7.2
UNIX	HP-UX 11.31	2007-02-15	2020-12-31	None
UNIX	Solaris 10.3	2005-01-31	2021-01-31	None
UNIX	Solaris 11.3	2011-11-09	***2034-11-30***	**7.3**

- Mac OS X 10.6 is last version for DISK distribution.
- Commercial UNIX is the ***longest End of Life (15-25 years)***.
- CentOS 5.0&5.11 has the oldest default Vim 7.0.
- We can assume the oldest default vim version is **Vim 7.0** (next 7.3 probably).

Standard Vim function

Interface

Interface is important for extension Vim function.

Date	Version	Interface
1998-02-19	5.0	perl, python
1998-08-24	5.2	tcl, cscope
1999-07-26	5.4	OLE
2001-09-27	6.0	ruby
2001-09-27	6.0	deubgger, workshop, sign
2003-06-01	6.2	Netbeans
2006-05-08	7.0	MzScheme
2010-08-15	7.3	lua, python3

- perl and python2 is supported since old Vim 5.0.
- Lua and Python3 is supported recently in Vim history.

Standard Plugin

`:help standard-pugin-list` or `$VIMRUNTIME/plugin/README.txt`

Version	Name	Description
7.0	getscriptPlugin.vim	Downloading latest version of Vim scripts
6.0	gzip.vim	Reading and writing compressed files
6.0	netrw.vim (explorer.vim)	Reading and writing files over a network/directory
6.0	rrhelper.vim	used for --remote-wait editing
6.2	tohtml.vim	convert a file with syntax highlighting to HTML
7.0	matchparen.vim	Highlight matching parens
7.0	spellfile.vim	download a spellfile when it's missing
7.0	tarPlugin.vim	Tar file explorer
7.0	vimballPlugin.vim	Create a self-installing Vim script
7.0	zipPlugin.vim	Zip archive explorer
8.0	logipat.vim	Logical operators on patterns

- Almost standard plugin bundled on Vim 6.0 and 7.0.
- Standard plugins is enabled by default.

Ref: Talk of Linda_pp@VimConf2013, Do You Know about Vim Runtime Files? // Speaker Dec

Macro

$VIMRUNTIME/macros/README.txt

Bundled Version	Name	Description
5.0 or older	hanoi	Macros that solve the tower of hanoi problem.
5.0 or older	life	Macros that run Conway's game of life.
5.0 or older	maze	Macros that solve a maze (amazing!).
5.0 or older	urm	Macros that simulate a simple computer: "Universal Register Machine"
6.0 or older	***less.vim, sh, bat***	make Vim work like less (or more)
5.0 or older	dvorak	Dvorak keyboard support; adds mappings
7.0	editexisting	when editing a file that is already edited with " another Vim instance, go to that Vim instance
6.0 or older	justify	justifying text.
6.0 or older	***matchit***	makes the % command work better
5.0 or older	shellmenu	menus for editing shell scripts in the GUI version.
5.0 or older	swapmouse	swap left and right mouse buttons

Since Vim 8.0, dvorak, editexisting, justify, matchit, shellmenu, swapmouse is moved $VIMRUTNIME/pack.

Macro is existing since old Vim release.

Macros is not enabled on default. If you want to use macro, try like following commands.

```
0001 | :source $VIMRUNTIME/<macro-name>.vim
```

```
0001 | :runtime macros/<macro-name>vim
```

Ref: talk of Linda_pp@VimConf2013, Do You Know about Vim Runtime Files? // Speaker Dec

Bundled Commands

Vim has 2 types of bundled commands.

1. Alias: ex (vim -e), vimdiff (vim -d), etc.
2. External commands: xxd, diff, etc.

Bundled commands is useful for less command OS (Windows).

Bundled Commands: Alias


```
:help starting
```

Alias	Option	Description
ex	vim -e	Start in Ex mode (see |Ex-mode|).
exim	vim -E	Start in improved Ex mode (see |Ex-mode|).
view	vim -R	Start in read-only mode (see |-R|).
gvim	vim -g	Start the GUI (see |gui|).
gex	vim -eg	Start the GUI in Ex mode.
gview	vim -Rg	Start the GUI in read-only mode.
rvim	vim -Z	Like "vim", but in restricted mode (see |-Z|).
rview	vim -RZ	Like "view", but in restricted mode.
rgvim	vim -gZ	Like "gvim", but in restricted mode.
rgview	vim -Rgz	Like "gview", but in restricted mode.
evim	vim -y	Easy Vim: set 'insertmode' (see |-y|).
eview	vim -yR	Like "evim" in read-only mode.
vimdiff	vim -d	Start in diff mode |diff-mode|.
gvimdiff	vim -gd	Start in diff mode |diff-mode|.

ex, gvim, vimdiff is useful alias especially.

Bundled Commands: External Commands

Official Vim archive for Windows (gvim*.exe) is here: ftp://ftp.vim.org/pub/vim/pc/

Windows Vim is installed `C:\Program Files (x86)\Vim\`. External commands have .exe extension.

Find external commands by following cmd.exe commands.

```
0001 dir /s /b "%ProgramFiles(x86)%\Vim\*.exe"
```

Command	Vim 5.0	Vim 6.0	Vim 7.0	Vim 8.0(.586)	Description
install.exe		○	○	○	Installer.
uninstal.exe		○	○	○	Uninstaller.
uninstall-gui.exe		○	○	○	Uninstaller with GUI.
vim.exe			○	○	CUI vim.
gvim.exe	○	○	○	○	GUI vim.
vimrun.exe		○	○	○	Vim interface for external commands on cmd.exe.
ctags.exe	○				Creating tags (DB for code definition location).
xxd.exe	○	○	○	○	Required for binary editing.
diff.exe			○	○	Required for ***vimdiff***.
tee.exe				○	Required for `:make` in progress output.

- ctags have not been bundled since Vim 6.0.
- tee.exe is bundled on Vim 8.0.
- **xxd.exe** and **diff.exe** is available for default Vim.

Tips for Compatible Vim

Concept for Compatible Vim

- Vim has many function for compatibility.
- Before using new function, ***check them***.
- Or setting option appropriately.

```
0001 if <is-enabled-new-feature>
0002     <use-new-feature-here>
0003 endif
0004 set <general-option>=<env1>,<env2>,<env3>
```

Topic of Compatible Tips

- Version
- Feature: has()
- Definition: exists()
- File Access
- Encodings

Version

Item	Description
v:version	Vim major & minor version (5.01 = 501).
has('patch-X.X.XXX')	Included patch (ex: has('patch-7.4.123')).

Patch is included next version. So sometimes we need to use `v:version` and `has('patch-X.X.XXX')`

Example

> *Patch 7.4.1952*
> *Problem: Cscope interface does not support finding assignments.*
> *Solution: Add the "a" command. (ppettina, closes #882)*
> *Files: runtime/doc/if_cscop.txt, src/if_cscope.c*

—*Vim: version8.txt*

```
0001 if has('cscope')
0002   set cscopequickfix=s-,g-,d-,c-,t-,e-,f-,i-
0003   if v:version >= 800 || has('patch-7.4.1952')
0004     let &cscopequickfix .= ',a-'
0005   endif
0006 endif
```

Feature: `has()`

`has()` is available for checking if interface, OS, GUI, compiled feature enabled (`:help featurelist`).

Item	Value
Interface	lua, mzscheme, perl, python, python3, ruby, tcl
GUI	gui_running, mouse
UNIX	unix
Mac	mac, macunix, osx
Windows	win16, win32, win64
Cygwin, MSYS(2)	win32unix

If you want to know detail about OS, use external commands by `system()`.

OS	Command
Linux	uname -s, cat /etc/os-release, lsb_release -a
Windows	VER

Example of `has()`


```
0001 """ Platform
0002 let s:IS_WINDOWS   = has('win64') || has('win32')   || has('win16')
0003 let s:IS_CYGWIN    = has('win32unix')
0004 let s:IS_MAC       = has('mac')   || has('macunix') || has('gui_macvim')
0005 let s:IS_LINUX     = has('unix')  && !s:IS_MAC      && !s:IS_CYGWIN
0006 let s:IS_WINDOWS_7 = s:IS_WINDOWS && system('VER') =~# 'Version 6.1'
0007
0008 "" mouse
0009 if has('mouse')
0010   set mouse=a
0011   set ttymouse=xterm2
0012 endif
```

Definition: `exists()`

`exists()` is available for checking if variable, option, function, event enabled.

Item	Description
&option-name	Vim option (only checks if it exits).
+option-name	Vim option that works.
$ENVNAME	environment variable.
****funcname***	built-in function or user defined function.
varname	internal variable.
:cmdname	Ex command: built-in command, user command.
:2match	The :2match command.
:3match	The :3match command.
#event	autocommand defined for this event.
#event#pattern	autocommand defined for this event and pattern..
#group	autocommand group exists.
#group#event	autocommand defined for this group and event.
#group#event#pattern	autocommand defined for this group, event and pattern.
##event	autocommand for this event is supported.

+option-name, *funcname, :cmdname, ##event is often used.

Example of `exists()`

In .vimrc, `exists('+option-name')` is enough for option compatibility.

New option needs to check by `exists()`.

```
0001 if exists('+packpath')
0002     set packpath+=~/.vim
0003 endif
0004
0005 if exists('##CmdlineEnter')
0006     autocmd CmdlineEnter * pwd
0007 endif
```

> *Release v8.0.1206: patch 8.0.1206: no autocmd for entering or leaving the command line · vim/vim*

File Access

If we want to use external exe or file, check it enabled.

Item	Description
executable({expr})	checks if an executable with the name {expr} exists.
exepath({expr})	return the full path for executable.
filereadable({file})	TRUE when a file with the name {file} exists, and can be read.
filewritable({file})	The result is a Number, which is 1 when a file with the name {file} exists, and can be written.
finddir({name}[, {path}[, {count}]])	Find directory {name} in {path}.
findfile({name}[, {path}[, {count}]])	Just like finddir(), but find a file instead of a directory.
shellescape({string} [, {special}])	Escape {string} for use as a shell command argument.
system({expr} [, {input}])	Get the output of the shell command {expr} as a string.
systemlist({expr} [, {input}])	Same as system(), but returns a List with lines (parts of output separated by NL) with NULs transformed into Nls.

executable, filereadable, shellescape, system is often used.

Example of File Access


```
0001 ## For :grep
0002 if executable('ag')
0003   set grepprg=ag\ --vimgrep\ $*
0004   set grepformat=%f:%l:%c:%m
0005 endif
```

```
0001 "" Add executable permission for shebang files
0002 autocmd BufWritePost * :call s:Add_execmod()
0003 function! s:Add_execmod()
0004   let s:line = getline(1)
0005   if strpart(s:line, 0, 2) == '#!'
0006     let s:IS_WINDOWS   = has('win64') || has('win32')   || has('win16')
0007     if s:IS_WINDOWS
0008       call system('icacls '    . shellescape(expand('%') . ' /grant ' . $USERNAME . ':(X)'))
0009     else
0010       call system('chmod +x -- ' . shellescape(expand('%')))
0011     endif
0012   endif
0013 endfunction
```

Encodings

- Opening file is basic and important as a text editor.
- Non English speaker needs multi byte characters.
- `.vimrc` has to support encoding in your country.

I'll talk about encoding here to end.

Vim Configuration for Encodings

`help: mbyte-encoding`

Option	Description
	Encoding options
encoding	Sets the character encoding used inside Vim (viminfo etc.).
termencoding	Encoding used for the terminal.
:scriptencoding	Specify the character encoding used in the script.
fileencoding	Sets the character encoding for the file of this buffer.
fileencodings	This is a list of character encodings considered when starting to edit an existing file.
	Other text options
ambiwidth	Tells Vim what to do with characters with East Asian Width Class Ambiguous.
bomb	When writing a file and fileencoding is a unicode variants, a BOM (Byte Order Mark) is prepended to the file.
endofline	When writing a file and this option is off and the 'binary' option is on, or 'fixeol' option is off, no <EOL> will be written for the last line in the file.
fixendofline	When writing a file and this option is on, <EOL> at the end of file will be restored if missing.
fileformat	This gives the <EOL> of the current buffer, which is used for reading/writing the buffer from/to a file. dos: <CR> <NL>, unix: <NL>, mac <CR>.
fileformats	This gives the end-of-line (<EOL>) formats that will be tried when starting to edit a new buffer and when reading a file into an existing buffer.

Basic Concept for Character Encoding

(Coded) character set (ex. Unicode)

↓↑

Character encoding (IANA) (ex. UTF-8)

↓↑

Byte (bit)

Example for Basic Concept

Character set	ASCII	Unicode	Unicode	JIS X 0201 JIS X 0208	JIS X 0208 + α	ASCII JIS X 0201 JIS X 0208 JIS X 0212	ASCII JIS X 0201 JIS X 0208
Character	A	あ	あ	あ	あ	あ	あ
	↓↑	↓↑	↓↑	↓↑	↓↑	↓↑	↓↑
Character encoding	US-ASCII, UTF-8, EUC-JP, Shift_JIS, ISO-2022-JP	UTF-8	UTF-16LE	Shift_JIS	Windows-31J (CP932)	EUC-JP	ISO-2022-JP
	↓↑	↓↑	↓↑	↓↑	↓↑	↓↑	↓↑
Byte (hex)	41	e3 81 82	42 30	82 a0	82 a0	a4 a2	1b 24 42 24 22

Typical Character Set

Character set	Including set	Bit	Byte	Amount	Description	Example
For English						
ASCII (ISO/IEC 646)		7	1	128	number, alphabet, sign.	123abcABC+-*/
Latin-1 (ISO/IEC 8859-1)	ASCII + α	8	1	256	ASCII + umlaut, accent，sign for EU.	£, À
For universe						
Unicode (ISO/IEC 10646, UCS)		32	4	1114112	One set for all of the world character.	
For Japanese						
JIS X 0201	ASCII modified + 1 byte Katakana	8	1	256	Initial Japanese set with ‾ modifying ASCII \→¥, ~→ ‾.	ｱ, ¥, ‾
JIS X 0208	JIS X 0211	8	2	6879	Basic Japanese set. including Kanji.	あ
JIS X 0211	Latin-1 subset	8	2	61	Control sequence.	
JIS X 0212		8	2	6067	Additional Kanji.	

Typical Encoding

Character encoding	Character set	Bit for encoding	Byte per char	ASCII compatible	BOM	Feature	Example
For English							
US-ASCII	ASCII	7	1	○			General PC character.
ISO-8859-1 (latin1)	ISO/IEC 8859-1	8	1	○		US-ASCII extended.	General English document.
For universe							
UTF-8	Unicode	8	2-6	○	○	ASCII compatible and supporting almost character.	General PC character.
UTF-16 (UCS-2)	Unicode	***16***	2, 4	×	○	UTF-16 is including UCS-2. For internal encoding.	Windows registry. Java, Javascript internal.
UTF-32 (UCS-4)	Unicode	***32***	4	×	○	Fixed byte. For internal encoding.	
For Japanese							
ISO-2022-JP	ASCII, JIS X 0201, JIS X 0208	7	1-2	○		Switch character set by escape sequence.	E-mail.
EUC-JP	ASCII, JIS X 0201, JIS X 0208 JIS X 0212	8	1-2	○		Switch a character by shift character.	Old UNIX like OS.
Shift_JIS	JIS X 0201, JIS X 0208	8	1-2	○		Historical Japanese default encoding.	Historical Japanese document.
Windows-31J (CP932, MS932)	JIS X 0208 + α (ex. ①, ㎞, ㎝)	8	1-2	○		***Microsoft customized Shift_JIS*** (Windows default Japanese set).	**Japanese Windows default.**

- Windows-31J (CP932) is almost super set for Shift_JIS.
- In software, use **CP932 instead of Shift_JIS**.

BOM (Byte Order Mark)

Multi byte encoding is depends on architecture (Big-endian, Little-endian).

Original Byte	Big-Endian	Little-Endian
12 34	12 34	34 12

UTF-16 and UTF-32 is encoding a character from multi bytes.

For detecting endian, append mark (BOM) to start of the file.

	UTF-16	UTF-32
BOM for Big Endian	FE FF	00 00 FE FF
BOM for Little Endian	FF FE	FF FE 00 00

No BOM means Big-endian.

UTF-8 is not depends on order of byte. UTF-8 BOM is only mark for easy detecting (no meaning order).

- **BOM is required for some program (Visual Studio, Microsoft Excel csv, etc.)**
- **But some program do not accpet BOM (shell script, bat file)**

In Vim, `set bomb` and `set nobomb` are available for appending/removing BOM only Unicode encoding.

Vim Specific Encoding

Vim supports many encoding. You can check `:help encoding-names` or `iconv -l`.

Following value is vim specific encoding names.

Value	Description
default	Default value of 'encoding', depends on environment.
japan	UNIX: euc-jp, Windows: cp932.
ucs-bom	Encoding for starting BOM (utf-8, utf-16, utf-32).

Can You Open These Files Correctly?

encoding-test.zip

(https://senooken.jp/public/20171104/encoding-test.zip)

encoing-test.zip is including following encoding and content.

utf-16le.txt	utf-8.txt	cp932.txt	euc-jp.txt	iso-2022-jp.txt	cp932-xsjis.txt
0001 | あア	0001 | あア	0001 | あア	0001 | あア	0001 | あ	0001 | あア①

If your .vimrc is compatible, you can open these files correctly.

(We can check following command for `fileencodings` test without .vimrc.)

```
0001 " vim -u NONE --cmd "set fencs=<encs>" file.txt
0002 vim -u NONE --cmd "set fencs=utf-8" cp932.txt
```

`fileencodings` for Japanese

- `fileencoding`***s*** is list for `fileencoding` (text file encoding).
- Checking is starting from first `fileencodings` item until success.
- Some encodings **misunderstand** other encoding. And some encoding is share of subset (ex. ASCII).
- **Order of `fileencodings` is important.**

Priority	Relation	Example of NG
1	ucs-bom >> other	
2	iso-2022-jp > utf-8, euc-jp, cp932	`vim -u NONE --cmd "set fencs=utf-8,iso-2022-jp" iso-2022-jp.txt`
3	euc-jp > cp932	`vim -u NONE --cmd "set fencs=cp932,euc-jp" euc-jp.txt`
4	utf-8 > cp932	`vim -u NONE --cmd "set fencs=cp932,utf-8" utf-8.txt`

```
0001 "" Example of fileencodings for all clear encoding-test.zip
0002 set fileencodings=ucs-bom,iso-2022-jp,utf-8,euc-jp,cp932
```

This setting is **not perfect**. If we try opening UTF-16 without BOM, encoding is failed. But almost good.

```
0001 "" Fix 'fileencoding' to use 'encoding' if the buffer only ASCII characters.
0002 autocmd BufReadPost *
0003   \   if &modifiable && !search('[^\x00-\x7F]', 'cnw')
0004   \ |   setlocal fileencoding=utf-8
0005   \ | endif
```

`fileformats`

- `fileformat` gives the `<EOL>` (end-of-line, line break).
  - dos: `<CR> <NL>`
  - unix: `<NL>`
  - mac: `<CR>`
- `fileformat`***`s`*** is list for `fileformat`.
- All list items of `fileformats` is checked automatically.
- Order of `fileformats` is not so important as `fileencodings`.
- First item is used for empty file (or all item mismatched file).

```
0001 set fileformats=unix,dos,mac
```

End of Line (EOL)

Vim has been **forcing to append line break** to end of file in writing text file for a long time.

For keeping default EOL, add `nofixendofline` option to `.vimrc` in Vim 8.0 (vim 7.4.785)

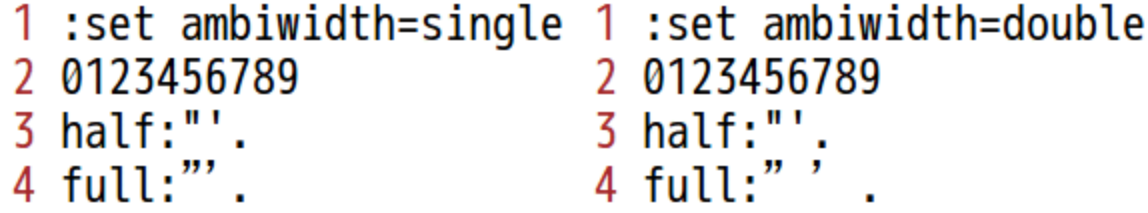
> *最後に改行がないファイルが作れない · Issue #152 · vim-jp/issues*

```
0001 "" Disabled to append LF automatically in writing
0002 if exists('+fixendofline')
0003     autocmd BufWritePre * setlocal nofixendofline
0004 endif
```

Then, `set endofline` and `set noendofline` is appending/removing EOL.

ambiwidth

- In east Asia, assign 1 byte character to half width, 2 bytes character to full width in practice.
- `ambiwidth` option operate these appearance.
- Default is `'single'`. For Japanese, `'double'` is recommended.




```
0001 set ambiwidth=double
```

statusline

It is useful for text file information on `statusline`.

Example

```
0001 "" Statusline
0002 set laststatus=2
0003 set statusline =[%n]%<%f    " File name
0004 set statusline+=\ %m%r%h%w  " Flag for modified, readonly, help buffer, preview
0005 set statusline+=%<%y        " filetype
0006 set statusline+=%{'['.(&fenc!=''?&fenc:&enc).':'.&ff.']'}  " Encoding
0007 set statusline+=%{&bomb?'[bomb]':''}   " BOM
0008 set statusline+=%{&eol?'':'[noeol]'}   " EOL
0009 set statusline+=[%04B]                 " Character code
0010 set statusline+=%=\ %4l/%4L\|%3v\|%4P  " Current position
```

```
  1 Vim
[1]half.txt [+][text][utf-8:unix][bomb][noeol][0056]    1/   1|  1| All
```

Summary

- Updating vs. Keeping
- Methodology for keeping: POSIXism
- Vim overview
    - hisotry
    - default version
    - bundled files
- Tips for compatibility
    - version, exists(), has()
    - File access
    - Encoding


Write Once,
Work
Anytime/Anywhere
posixism.org