Preparation

`git clone https://github.com/aswen/vim-workshop.git`
or
`github.com/aswen/vim-workshop/archive/master.zip`
`(bit.ly/16s4tMQ)`

```
vim-workshop/
|-files/ (excercises)
|-presentation/ (this presentation and handouts.pdf)
|-vim-config/
|-startvim <= Script to startup Vim
```

Not yet installed Vim? => www.vim.org => download
No Internet, ask me for usb stick.

Alexander Swen | vim@swen.nu

Based on workshop created together with
Joël Stemmer for Nedap

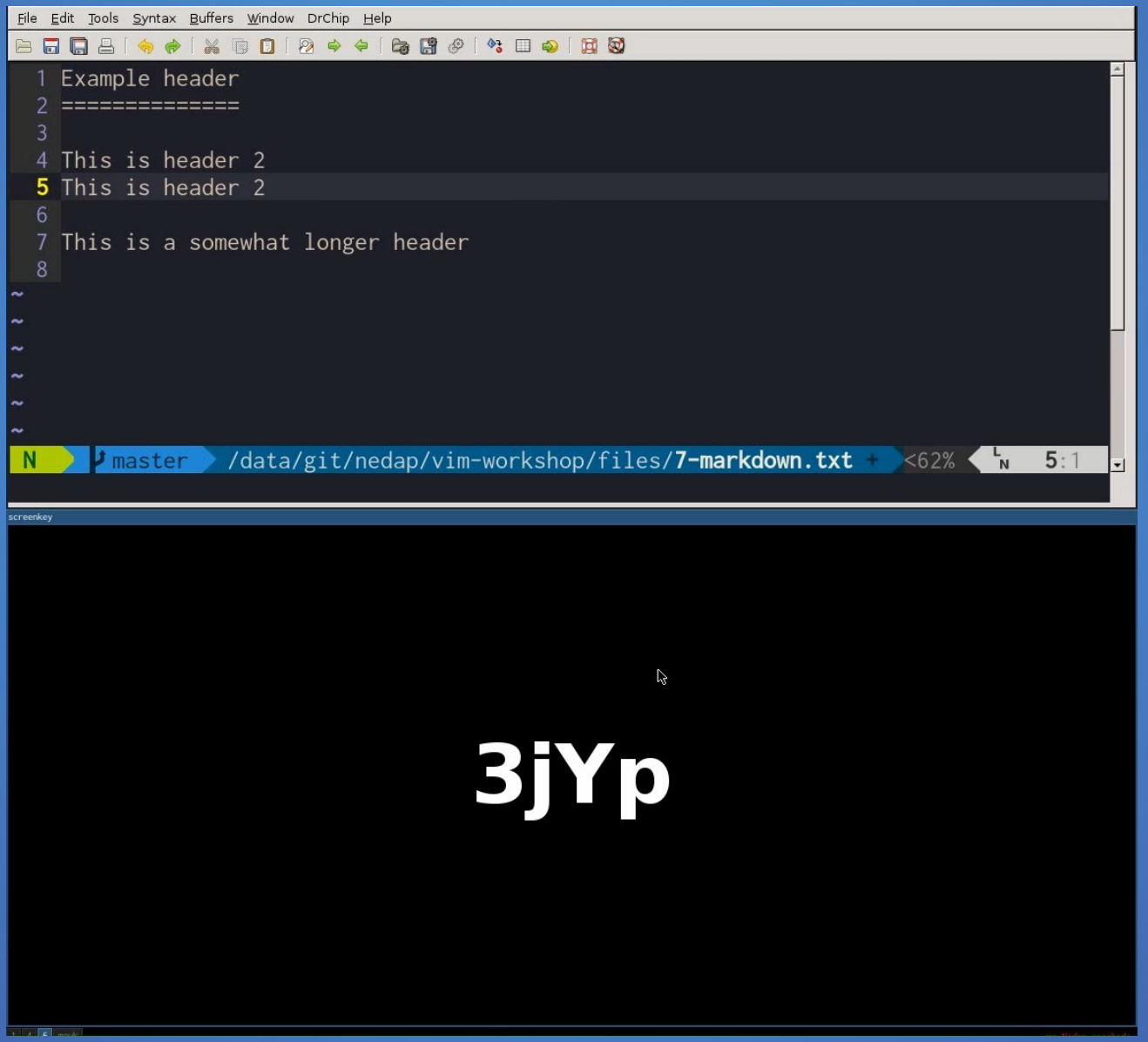
Why Vim?

```
Example header
==============

This is header 2
```

Why Vim?

```
Example header
==============

This is header 2
================
```

Why Vim?

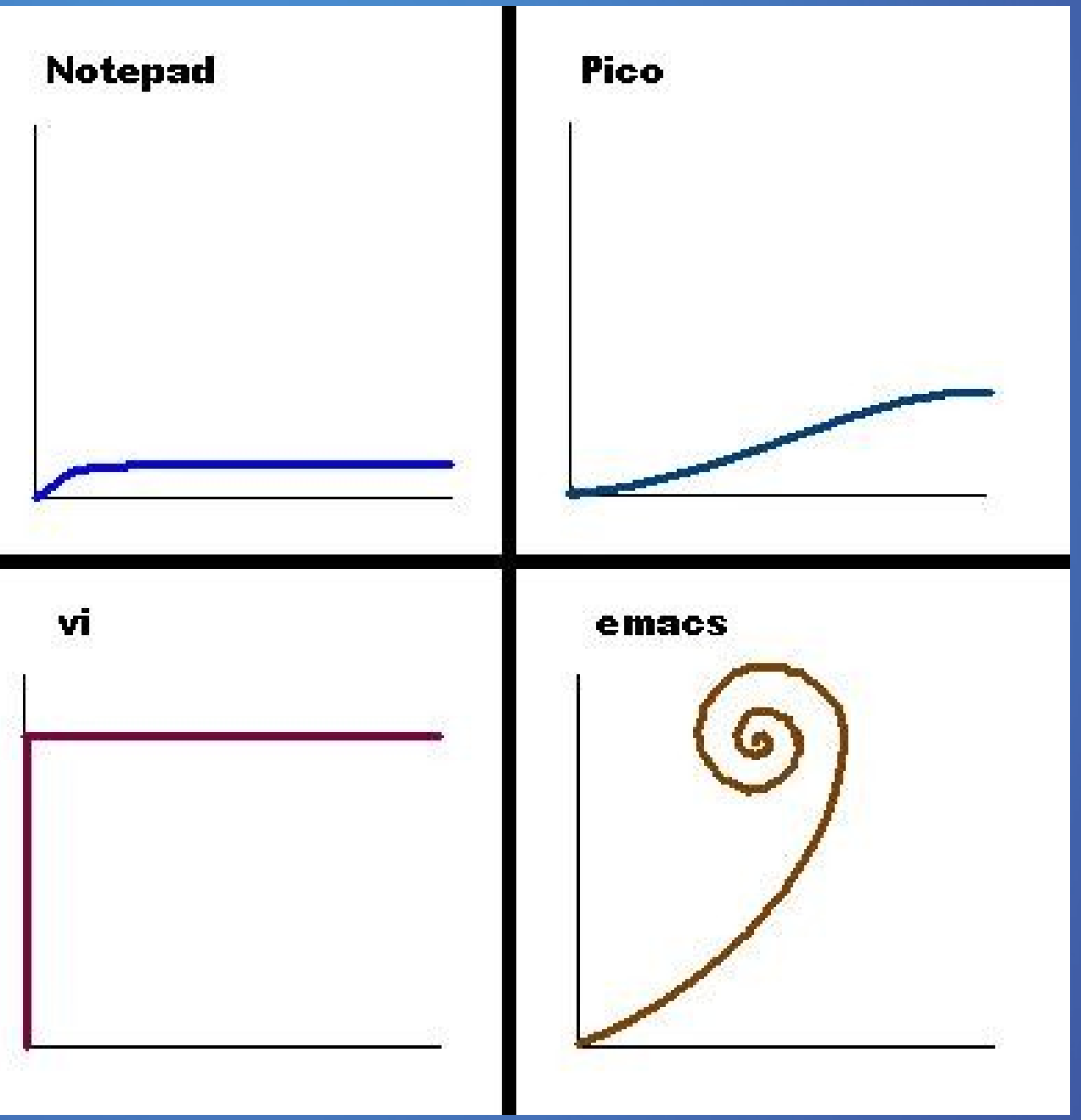

File Edit Tools Syntax Buffers Window DrChip Help
1 Example header
2 ==============
3
4 This is header 2
5 This is header 2
6
7 This is a somewhat longer header
8
N master /data/git/nedap/vim-workshop/files/7-markdown.txt + <62% 5:1
screenkey
3jYp

Why Vim?

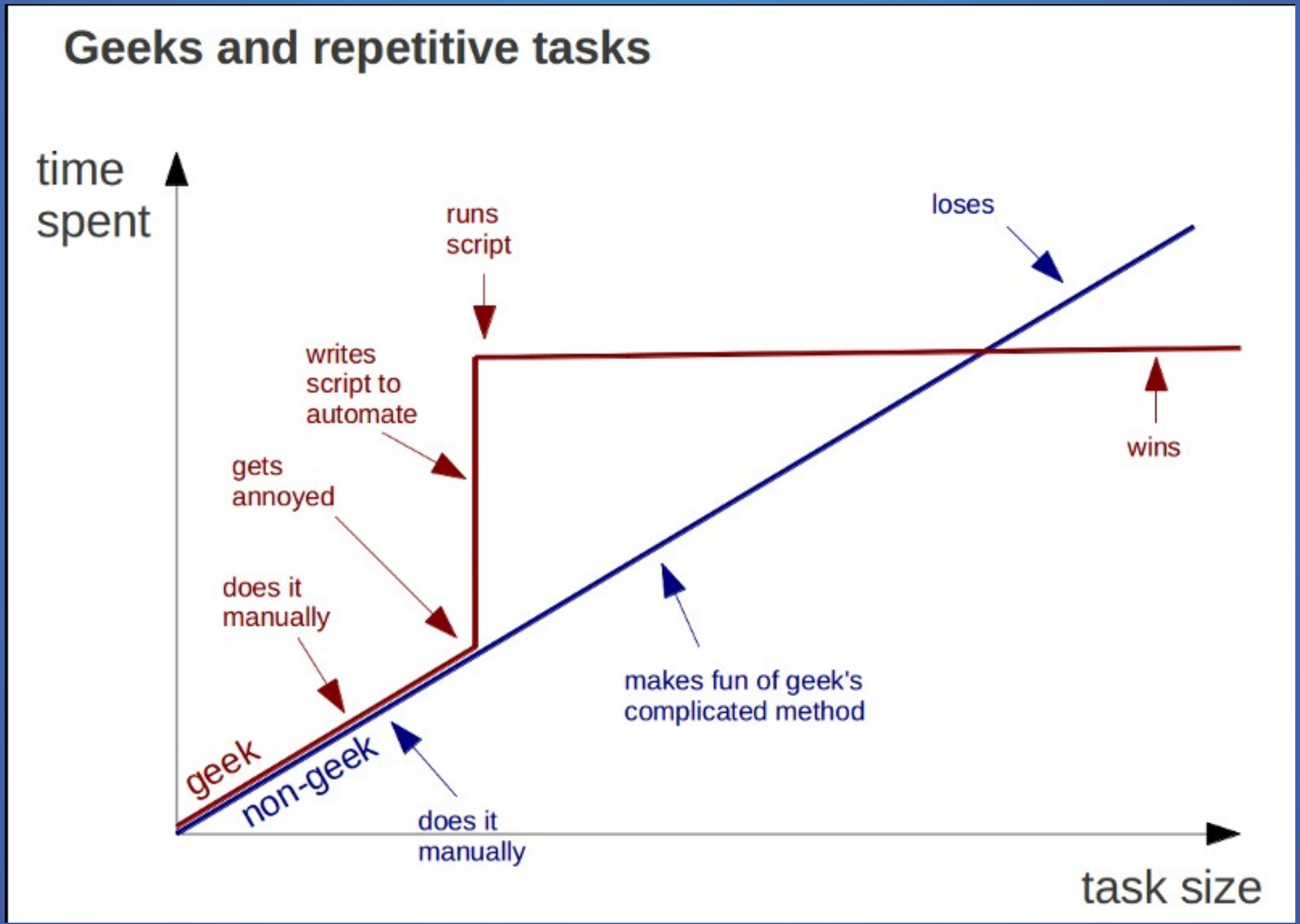
```
Example header
==============

This is header 2
================

This is a somewhat longer header...
```

Why Vim?

Classical learning curves for some common editors



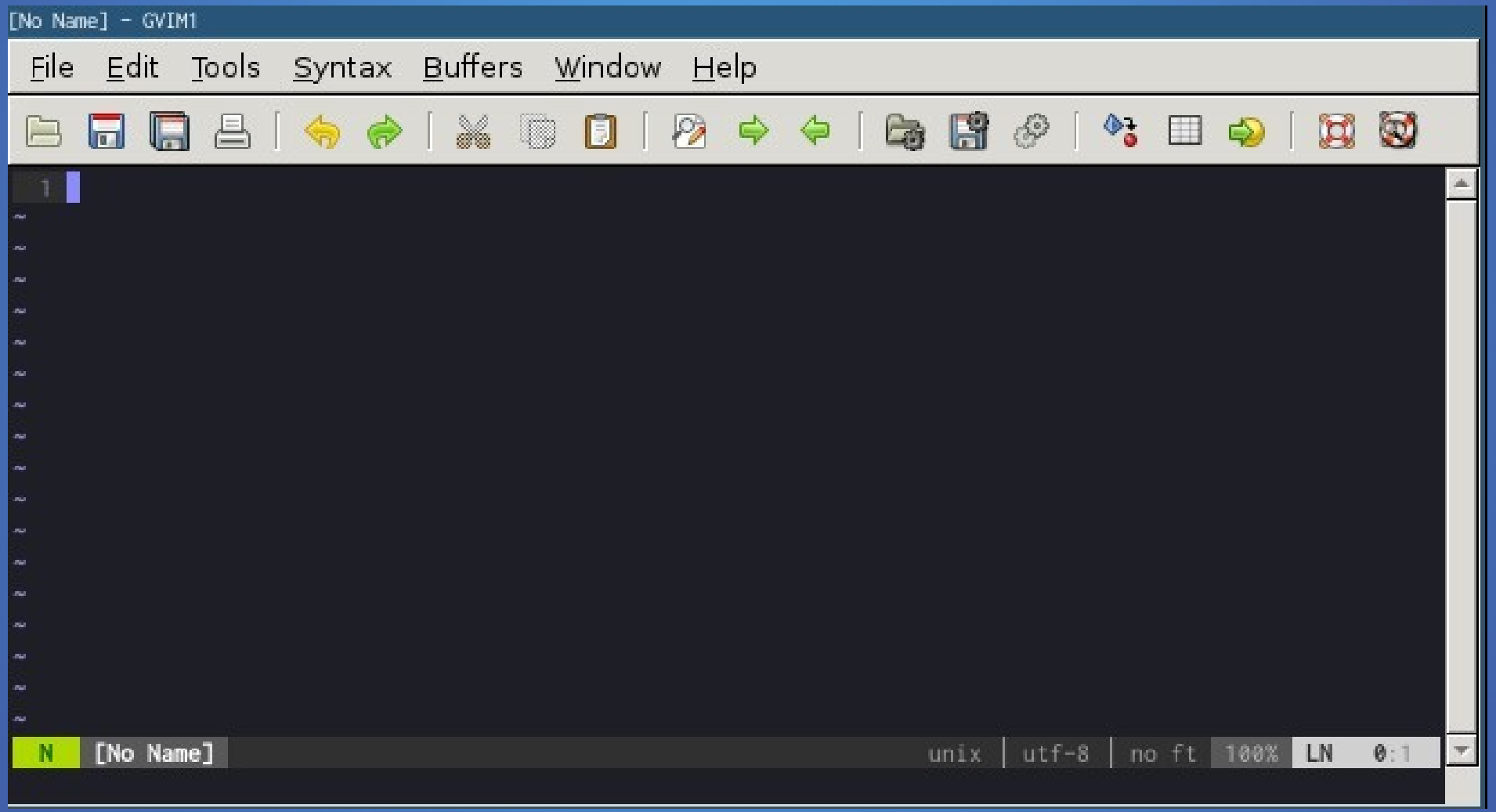
Why Vim?


Geeks and repetitive tasks
time spent
runs script
loses
writes script to automate
wins
gets annoyed
does it manually
makes fun of geek's complicated method
geek
non-geek
does it manually
task size

Brief history of Vim

Brief history of Vim

Vim Gui


[No Name] - GVIM1
File Edit Tools Syntax Buffers Window Help
1
N [No Name] unix | utf-8 | no ft 100% LN 0:1

Don't Panic

`ESC`	Back to normal mode
`:q!`	Quit discarding changes
`:wq`	Write changes & quit (or ZZ, or :x)
`:e [file]`	Edit [file]
`:e!`	Reloads file
`:h`	Help!

Modes

Vim Modes

`Normal mode (ESC)`
- Move cursor/view
- Text manipulation
- Ex mode commands (:<cmd>)
- Search (/{string})

`Insert mode (i)`
- Input text

`Visual mode (v)`
- Select text

Vim Modes

NORMAL mode	ESC	
INSERT mode	i	**i**nsert
	I	**I**nsert @BOW (begin of 1st non-whitespace character)
	a	**a**ppend
	A	**A**ppend @EOL (End Of Line)
	o	**o**pen line
	O	**O**pen line above
VISUAL mode	v, V or CTRL+v	

Undo/ Redo

u	**u**ndo
CTRL+r	**r**edo

Repeat actions

Repeat actions *N* times

`#<action> repeat <action>` # times

Example:

`4l` moves cursor 4 places to the right

`4i{some text }ESC` creates:

`some text some text some text some text`

Motions

Motions

h	j	k	l
←	↓	↑	→

Motions

h j k l

← ↓ ↑ →

Exercise

Startup vim (./startvim)

```
i f f b r ESC
a a ESC
a e ESC
3 h
i e ESC . (a dot)
l a SPACE ESC
A k ESC
I c o ESC
/a ENTER
xp
```

	reference
`i`	insert
`a`	append
`hjkl`	left,dn,up,right
`A`	Append *eol*
`.`	Repeat cmd
`I`	Insert *bow*
`/a`	Search “a”
`xp`	Delete/paste

Coffee break

More motions

HOME	0	begin of line (*bol*)
	^	1st non-whitespace
END	$	end of line (*eol*)
	gg	begin of file (*bof*)
	G	end of file (*eof*)

More motions

PAGEUP	CTRL+f	Page **f**orward
PAGEDOWN	CTRL+b	Page **b**ackward
	{ }	“Paragraph” up/down*
	zz	Center view on current {object}

*= Paragraph actually means “first empty line before/after this paragraph.

More motions

`w`	next **w**ord
`e`	**e**nd of word
`b`	**b**eginning of word
`:22`	goto line 22

More motions

w	next **w**ord
e	**e**nd of word
b	**b**eginning of word

These “bew” commands may not do what you expect:

More motions

Up-to-date

Use “BEW” instead

Operators

Operators

c	**c**hange (has to be followed by motion)
C	**C**hange line from cursor to *eol*
cc	**c**hange entire line
d	**d**elete (has to be followed by a motion)
D	**D**elete from cursor to *eol*
dd	**d**elete entire line

Operator + Motion

Operator + Motion

`c w`	**c**hange to end of **w**ord
`d b`	**d**elete to **b**egin of word (*bow*)
`c ^`	**c**hange to *bol*
`d G`	**d**elete to *eof*

Exercise

Startup vim (./startvim)

– open files/1-lorem.txt
– go to 3rd paragraph
– change 3rd word to “bla”
– change the following 4 words to “blabla”
– delete the 6th (empty) line
– go to the end of the 7th word of 4 paragraphs down and change from there to the beginning of that line to “this is fun”.

reference

`:e [file]`	Open [file]
`{ }`	Prev. / next paragraph
`c4w`	Change 4 words
`c`	change
`c^`	Change to *bol*
`dd`	Delete line
`4}`	Move down 4 paragraphs
`d G`	Delete from cursor to *eof*.

Even more motions

`f {char}`	**f**ind character
`t {char}`	**t**o/till character

Even more motions

`f {char}`	**f**ind character
`t {char}`	**t**o/till character
`;`	repeat
`,`	reverse

Operator + motion

`c f {char}`	**c**hange **f**ind character (incl that char)
`d t {char}`	**d**elete **t**o/till character
`v f {char}`	**v**isual select **f**ind character (incl that char)
`c 2 t {char}`	**c**hange **t**o/till 2nd character

Exercise

Open files/4-boilerplate.html

(Only use the motions and operators we learned so far)

– line 16: select navigation/extensions and swap them

– line 18: change deck.hashes.css to deck.hash.css

– line 82: same as 18

reference

`c f {char}`	**c**hange **f**ind character (incl that char)
`d t {char}`	**d**elete **t**o character
`v f {char}`	Select **f**ind character (incl that char)

Searching

Searching

`/{string}`	Search string (may be regex)
`n`	next occurrence
`SHIFT+n (N)`	previous occurrence (backwards)

Operator + search

`c / {string}`	Change to string
`d / {string}`	Delete to string
`v / {string}`	Select till 1st char of string!

Exercise

Open files/4-boilerplate.html

(Only use the motions and operators we learned so far)

– line 16: select navigation/extensions and swap them

– line 18: change deck.hashes.css to deck.hash.css

– line 82: same as 1

reference

`/{string}`	Search {string}
n	Next occurrance

Cut/Copy/Paste & visual mode

Cut/Yank(copy)/Paste

y w	**y**ank **w**ord
y y	**y**ank current line
d w	**d**elete **w**ord
p	**p**aste after
SHIFT+p (P)	**P**aste before

Visual mode

`v`	**v**isual mode
`SHIFT+v` (`V`)	**V**isual line mode
`CTRL+v`	**v**isual block mode

Visual mode + motions

`v w`	**v**isual select **w**ord
`SHIFT+v 3j` `(V)`	**V**isual select 4 lines
`CTRL+v G`	**v**isual select a block to eof

Visual mode + operators

c	**c**hange selection
d	**d**elete selection
y	**y**ank (copy) selection
r	**r**eplace each character in selection
p	**p**aste over selection
:	Ex mode command (example: :wq and :e [file])

Exercise

Open files/6-digits.txt

– Remove lines 1, 3 and last
– Make a csv file

(you don't have to save the file)

	reference
v	**v**isual mode
SHIFT+v	**V**isual line mode
CTRL+v	**v**isual block mode
c	**c**hange selection
d	**d**elete selection
y	**y**ank (copy) selection
r	**r**eplace selection
p	**p**aste over selection
:	Ex command mode (e.g. :s/search/replace/)

Search/replace

`:s/before/after/`	Replaces the first occurance of the word “before” with “after” on this line.
`:s/before/after/g`	Replaces each occurance of the word “before” with after on this line or selection (global)
`:%s?/usr/lib?/usr/lib64?g`	Entire file, ? is allowed as well as a delimiter
`:24,27:s<true<false<g`	On lines 24-27, < is also a valid delimiter
`&`	Repeat last search/replace

Exercise

Open files/6-digits.txt

– Remove lines 1, 3 and last
– Make a csv file (using search/replace)

reference

```
:s/search/replace/
:%s?search?replace?g
```

Exercise

Open files/6-digits.txt

– Remove lines 1, 3 and last
– Make a csv file (using search/replace)

reference

```
:s/search/replace/
:%s?search?replace?g
```

```
:%s?\D\+\(\d\+\)\D\+\(\d\+\)\D\+?\1,\2?
```

Text objects

Text objects

`i`	**i**n/inner
`a`	**a**ll/around

More text objects

“ ’ ` ([{ }])	Delimiters (ci”)
w	**w**ord
s	**s**entence
p	**p**aragraph
t	**t**ag (bla)

Operator + text objects

`c i “`	**c**hange **i**nner string
`d i t`	**d**elete **i**nner **t**ag
`y a p`	**y**ank **a**ll **p**aragraph
`v a s`	**v**isual select **a**ll **s**entence
`c a w`	**c**hange **a**ll **w**ord

Fun for if we have time left...

Fun for if we have time left:

vimrc

buffers

windows

plugins

Vimrc

`:so %`	Reload vimrc while editing
`:so $MYVIMRC`	General reload
`repo:/files/(g)vimrc-alex`	My vimrc... bit messy. (g => gvimrc => if/else thing works only half afaik)

Tip: store your vimrc in a git repo
like so:
`https://github.com/aswen/dotvim`

buffers

\ b	open buffer explorer
ENTER	switch to selected buffer
d	delete buffer (close file)

buffers

\ b	open buffer explorer
ENTER	switch to selected buffer
d	delete buffer (close file)

windows

`CTRL + w s`	**w**indow **s**plit
`CTRL + w  v`	**w**indow **v**ertical split
`CTRL + w  q`	**w**indow **q**uit
`CTRL + w  h j k l`	**w**indow **f**ocus

plugins

Pathogen (github.com/tpope)

/bundle

```
git submodule add ...
git submodule init
git submodule update
```

Tool to make it easy to manage plugins

Git clone > bundle

Easier to use git submodules.
Easier updates.

Vim in other software

`More/less/man`	Motions and search
Vimium/vrome	Browser plugin for GC
pentadactyl	Browser plugin for FF
Bash/ksh (set -o vi)	Vim cmd line!
Zsh (bindkey -v)	Vim cmd line!

Plugins for eclipse and netbeans (etc)

vifm	Vim file manager

That's all

Alexander Swen
questions and feedback: vim@swen.nu
Irc://irc.freenode.org/vim: aswen

You can hire me to do this for your colleagues!

www.vim.org

Drew Neil: vimcasts.org
(he wrote the book “Practical Vim”)

vimcheatsheet.com

vimgolf.com
vim-adventures.com

Tim Pope: github.com/tpope

This workshop has been created with Joël Stemmer
for our colleagues at Nedap