

CAN YOU



-NISHAN PANTHA
(PARADOX)

WHAT IS VIM?

NOT THIS



THIS...



THIS...

```

NERD tree 1- (~/.workspace/my_project/geekhner) - VIM 162x42
* Press ? for help
.. (up a dir)
- workspace/my_project/geekhner/
- deploy/
- assets/
- blog/
- images/
- javascripts/
- plugins/
- public/
- robots_backup/
- sass/
- source/
- stylesheets/
- _config.yml
- atom.xml
- CHANGELOG.markdown
- config.rb
- config.ru
- favicon.png
- Gemfile
- Gemfile.lock
- index.html
- note.txt
- Rakefile
- README.markdown
- sitemap.xml

5 function prettyDate(time) {
6   if (navigator.appName === 'Microsoft Internet Explorer') {
7     return "<span><span></span>"; // because IE date parsing isn't fun.
8   }
9   var say = {
10     just now: "now",
11     minute ago: "1n",
12     minutes ago: "m",
13     hour ago: "1h",
14     hours ago: "h",
15     yesterday: "1d",
16     days ago: "d",
17     last week: "1w",
18     weeks ago: "w"
19   };
20
21   var current_date = new Date(),
22       current_date_time = current_date.getTime(),
23       current_date_full = current_date_time + (1 * 86400),
24       date = new Date(time),
25       diff = ((current_date_full - date.getTime()) / 1000),
26       day_diff = Math.floor(diff / 86400);
27
28   if (isNaN(day_diff) || day_diff < 0) { return "<span><span></span>"; }
29
30   return day_diff === 0 && (
31     diff < 60 && say.just now ||
32     diff < 120 && say.minute ago ||
33     diff < 3600 && Math.floor(diff / 60) + say.minutes ago ||
34     diff < 7200 && say.hour ago ||
35     diff < 86400 && Math.floor(diff / 3600) + say.hours ago ||
36     day_diff === 1 && say.yesterday ||
37     day_diff < 7 && day_diff + say.days ago ||
38     day_diff === 7 && say.last week ||
39     day_diff > 7 && Math.ceil(day_diff / 7) + say.weeks ago;
40 }
41
42 function linkifyTweet(text, url) {
43   // Linkify urls, usernames, hashtags
44   text = text.replace(/(https?:\/\/\w+)?(?!\/\w+):?&#x2F;?%[\w\+]/gi, '<a href="$1$2">$2</a>')

```

THIS...

```

1 # -*- coding: utf-8 -*-
2
3 class Database:
4     """
5     Class to represent a database stored inside a rasdaemon server
6     """
7
8     def __init__(self, connection, name):
9         """
10         Constructor for the Database Class
11         (param connection) the connection object for the rasdaemon server
12         (param str name) the name of the database
13         """
14         self.connection = connection
15         self.name = name
16         self.rasmgr_db = None
17         self.rasrvr_db = None
18         self.channel = None
19         self.stub = None
20         self.rasrvr_keep_alive_running = None
21         self._keep_alive_thread = None
22         self.open()
23
24     def open(self):
25         """
26         Opens a connection to the RasServer on which the database is stored
27         and starts sending the keep alive messages to the RasServer. Also, stops
28         sending keep alive messages to the RasManager in case they are on the
29         same machine.
30         """
31         self.rasmgr_db = rasmgr_open_db(self.connection.stub, self.connection.session.clientUID,
32                                         self.connection.session.clientId, self.name)
33         if self.rasmgr_db.dbSessionId == self.connection.session.clientUID:
34             self.connection._stop_keep_alive()
35         self.channel = implementations.insecure_channel(self.rasmgr_db.serverHostName, self.rasmgr_db)
36         self.stub = rasrvr_beta_create_ClientRasrvrService_stub(self.channel)
37         self.rasrvr_db = rasrvr_open_db(self.stub, self.connection.session.clientId, self.name)
38         self._keep_alive()
39
40     def close(self):
41         """
42         Closes the connection to RasServer and RasManager. Also, stops
43         sending the keep alive messages to the RasServer.
44         """
45         self._stop_keep_alive()
46         rasrvr_close_db(self.stub, self.connection.session.clientId)
47         rasmgr_close_db(self.connection.stub, self.connection.session.clientUID, self.connection
48                        self.rasmgr_db.dbSessionId)
49
50     def create(self):
51         """
52         Method for creating a collection
53         """
54         raise NotImplementedError("Sorry, not implemented yet")
55
56     def destroy(self):
57         """
58         Method for destroying a collection
59         """
60         raise NotImplementedError("Sorry, not implemented yet")
61
62     def transaction(self, run=False):
63         """
64         Method for starting a transaction
65         """
66         raise NotImplementedError("Sorry, not implemented yet")
67
68     def commit(self):
69         """
70         Method for committing a transaction
71         """
72         raise NotImplementedError("Sorry, not implemented yet")
73
74     def rollback(self):
75         """
76         Method for rolling back a transaction
77         """
78         raise NotImplementedError("Sorry, not implemented yet")
79
80     def __del__(self):
81         """
82         Destructor for the Database Class
83         """
84         self.close()
85
86     def __str__(self):
87         """
88         String representation of the Database Class
89         """
90         return "Database: " + self.name
91
92     def __repr__(self):
93         """
94         String representation of the Database Class
95         """
96         return "Database: " + self.name
97
98     def __eq__(self, other):
99         """
100         Equality comparison for the Database Class
101         """
102         return self.name == other.name
103
104     def __neq__(self, other):
105         """
106         Inequality comparison for the Database Class
107         """
108         return self.name != other.name
109
110     def __lt__(self, other):
111         """
112         Less than comparison for the Database Class
113         """
114         return self.name < other.name
115
116     def __gt__(self, other):
117         """
118         Greater than comparison for the Database Class
119         """
120         return self.name > other.name
121
122     def __le__(self, other):
123         """
124         Less than or equal comparison for the Database Class
125         """
126         return self.name <= other.name
127
128     def __ge__(self, other):
129         """
130         Greater than or equal comparison for the Database Class
131         """
132         return self.name >= other.name
133
134     def __hash__(self):
135         """
136         Hash function for the Database Class
137         """
138         return hash(self.name)
139
140     def __getitem__(self, key):
141         """
142         Item access for the Database Class
143         """
144         return self.__getitem__(key)
145
146     def __setitem__(self, key, value):
147         """
148         Item assignment for the Database Class
149         """
150         return self.__setitem__(key, value)
151
152     def __delitem__(self, key):
153         """
154         Item deletion for the Database Class
155         """
156         return self.__delitem__(key)
157
158     def __contains__(self, key):
159         """
160         Contains check for the Database Class
161         """
162         return self.__contains__(key)
163
164     def __iter__(self):
165         """
166         Iteration for the Database Class
167         """
168         return self.__iter__()
169
170     def __len__(self):
171         """
172         Length of the Database Class
173         """
174         return self.__len__()
175
176     def __bool__(self):
177         """
178         Boolean value of the Database Class
179         """
180         return self.__bool__()
181
182     def __nonzero__(self):
183         """
184         Non-zero value of the Database Class
185         """
186         return self.__nonzero__()
187
188     def __unicode__(self):
189         """
190         Unicode representation of the Database Class
191         """
192         return self.__unicode__()
193
194     def __bytes__(self):
195         """
196         Bytes representation of the Database Class
197         """
198         return self.__bytes__()
199
200     def __float__(self):
201         """
202         Float representation of the Database Class
203         """
204         return self.__float__()
205
206     def __int__(self):
207         """
208         Integer representation of the Database Class
209         """
210         return self.__int__()
211
212     def __long__(self):
213         """
214         Long representation of the Database Class
215         """
216         return self.__long__()
217
218     def __complex__(self):
219         """
220         Complex representation of the Database Class
221         """
222         return self.__complex__()
223
224     def __abs__(self):
225         """
226         Absolute value of the Database Class
227         """
228         return self.__abs__()
229
230     def __add__(self, other):
231         """
232         Addition of the Database Class
233         """
234         return self.__add__(other)
235
236     def __sub__(self, other):
237         """
238         Subtraction of the Database Class
239         """
240         return self.__sub__(other)
241
242     def __mul__(self, other):
243         """
244         Multiplication of the Database Class
245         """
246         return self.__mul__(other)
247
248     def __div__(self, other):
249         """
250         Division of the Database Class
251         """
252         return self.__div__(other)
253
254     def __mod__(self, other):
255         """
256         Modulus of the Database Class
257         """
258         return self.__mod__(other)
259
260     def __pow__(self, other):
261         """
262         Power of the Database Class
263         """
264         return self.__pow__(other)
265
266     def __divmod__(self, other):
267         """
268         Divmod of the Database Class
269         """
270         return self.__divmod__(other)
271
272     def __radd__(self, other):
273         """
274         Right addition of the Database Class
275         """
276         return self.__radd__(other)
277
278     def __rsub__(self, other):
279         """
280         Right subtraction of the Database Class
281         """
282         return self.__rsub__(other)
283
284     def __rmul__(self, other):
285         """
286         Right multiplication of the Database Class
287         """
288         return self.__rmul__(other)
289
290     def __rdiv__(self, other):
291         """
292         Right division of the Database Class
293         """
294         return self.__rdiv__(other)
295
296     def __rmod__(self, other):
297         """
298         Right modulus of the Database Class
299         """
300         return self.__rmod__(other)
301
302     def __rpow__(self, other):
303         """
304         Right power of the Database Class
305         """
306         return self.__rpow__(other)
307
308     def __rdivmod__(self, other):
309         """
310         Right divmod of the Database Class
311         """
312         return self.__rdivmod__(other)
313
314     def __neg__(self):
315         """
316         Negation of the Database Class
317         """
318         return self.__neg__()
319
320     def __pos__(self):
321         """
322         Positive of the Database Class
323         """
324         return self.__pos__()
325
326     def __neg__(self):
327         """
328         Negation of the Database Class
329         """
330         return self.__neg__()
331
320     def __pos__(self):
332         """
333         Positive of the Database Class
334         """
335         return self.__pos__()
336
337     def __abs__(self):
338         """
339         Absolute value of the Database Class
340         """
341         return self.__abs__()
342
343     def __add__(self, other):
344         """
345         Addition of the Database Class
346         """
347         return self.__add__(other)
348
349     def __sub__(self, other):
350         """
351         Subtraction of the Database Class
352         """
353         return self.__sub__(other)
354
355     def __mul__(self, other):
356         """
357         Multiplication of the Database Class
358         """
359         return self.__mul__(other)
360
361     def __div__(self, other):
362         """
363         Division of the Database Class
364         """
365         return self.__div__(other)
366
367     def __mod__(self, other):
368         """
369         Modulus of the Database Class
370         """
371         return self.__mod__(other)
372
373     def __pow__(self, other):
374         """
375         Power of the Database Class
376         """
377         return self.__pow__(other)
378
379     def __divmod__(self, other):
380         """
381         Divmod of the Database Class
382         """
383         return self.__divmod__(other)
384
385     def __radd__(self, other):
386         """
387         Right addition of the Database Class
388         """
389         return self.__radd__(other)
390
389     def __rsub__(self, other):
391         """
392         Right subtraction of the Database Class
393         """
394         return self.__rsub__(other)
395
396     def __rmul__(self, other):
397         """
398         Right multiplication of the Database Class
399         """
400         return self.__rmul__(other)
401
402     def __rdiv__(self, other):
403         """
404         Right division of the Database Class
405         """
406         return self.__rdiv__(other)
407
408     def __rmod__(self, other):
409         """
410         Right modulus of the Database Class
411         """
412         return self.__rmod__(other)
413
414     def __rpow__(self, other):
415         """
416         Right power of the Database Class
417         """
418         return self.__rpow__(other)
419
420     def __rdivmod__(self, other):
421         """
422         Right divmod of the Database Class
423         """
424         return self.__rdivmod__(other)
425
426     def __neg__(self):
427         """
428         Negation of the Database Class
429         """
430         return self.__neg__()
431
432     def __pos__(self):
433         """
434         Positive of the Database Class
435         """
436         return self.__pos__()
437
438     def __abs__(self):
439         """
440         Absolute value of the Database Class
441         """
442         return self.__abs__()
443
444     def __add__(self, other):
445         """
446         Addition of the Database Class
447         """
448         return self.__add__(other)
449
450     def __sub__(self, other):
451         """
452         Subtraction of the Database Class
453         """
454         return self.__sub__(other)
455
456     def __mul__(self, other):
457         """
458         Multiplication of the Database Class
459         """
460         return self.__mul__(other)
461
462     def __div__(self, other):
463         """
464         Division of the Database Class
465         """
4
```

...AND THIS...



TEXT EDITOR

YAY...

VI

IMPROVED

TEXT BASED USER INTERFACE

WHY?

BUT



WHY?

TYPE LESS, DO MORE

REMOVE REDUNDANT MOVEMENT

INCREASED CODING SPEED

AVAILABLE EVERYWHERE

CUSTOMIZATION

CUSTOMIZATION

plugins

PLUGIN MANAGER

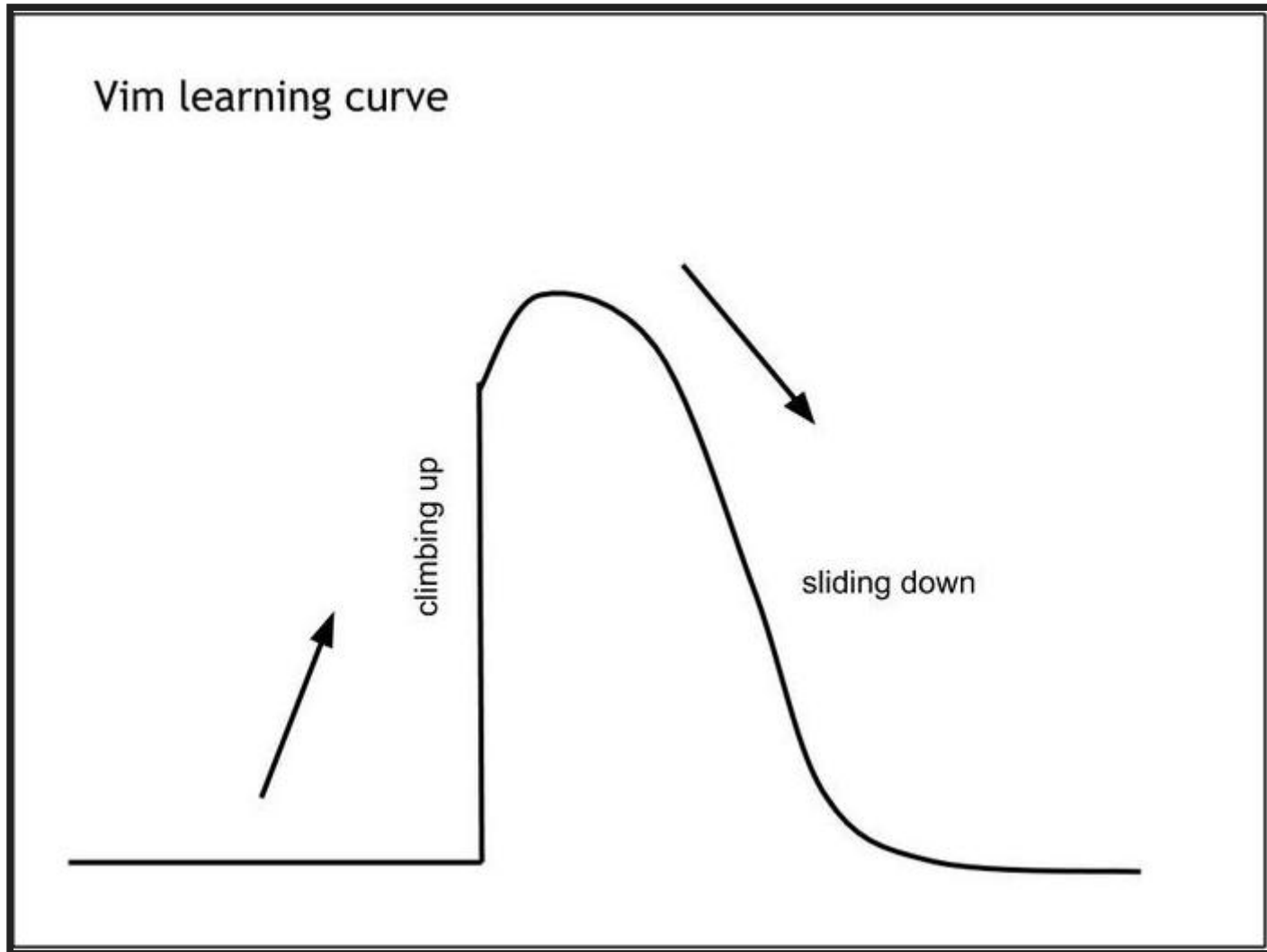
vundle

vim-plug

VIMRC

Unleash the power within

LEARNING CURVE



LET'S GET STARTED

MODAL BEHAVIOUR

3 POPULAR MODES

INSERT MODE

NORMAL MODE

VISUAL MODE

INSERT MODE

SAME AS ANY OTHER TEXT EDITOR

NORMAL MODE

ESCAPE TO NORMAL



NORMAL IS ABNORMAL

WHAT CAN YOU DO?

navigation

change settings

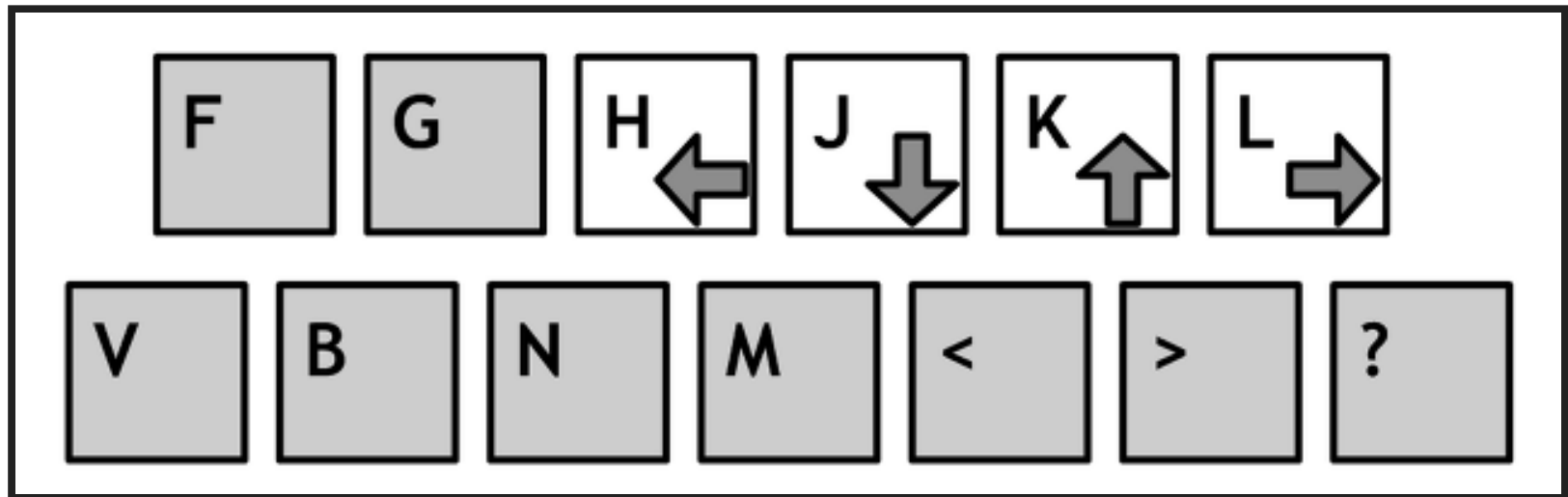
split screens and windows

load, save

YOU CAN DO ANYTHING

BASIC NAVIGATION

MOVE WITHOUT ARROWS



WORDWISE MOVEMENT

w : forward to beginning of word

b : backward to beginning of word

e : move to end of word

LINEWISE MOVEMENT

<number> gg : goto line no. number

gg : goto first line

G : goto end of the file

LINEWISE MOVEMENT

0 : move to beginning of the current line

\$: move to end of the current line

PAGEWISE MOVEMENT

CTRL-B : move one page up

CTRL-F : move one page down

DELETION

x : delete a single character under cursor

dd : delete current line

d <other motion> : delete with other motion

r <character> : replace current character by
character

MULTIPLY NAVIGATION

2h : move left by two characters

3l : move right by 3 characters

5j : move down by 5 lines

7k : move up by 7 lines

MULTIPLY NAVIGATION

2w : move forward 2 words

3b : move backward 3 words

d3w : delete next 3 words

d3b : delete next 3 words

MULTIPLE DELETION

d3w : delete next 3 words

d3b : delete next 3 words

NORMAL MODE

COPY/PASTE

yy or YY : copy(yank) current line

p : paste before the cursor

P : paste after the cursor

CUT

ANY DELETE MOTION

x / d / dd

ADD OTHER MOTION

SEARCH

/

goto search mode

forward slash followed by letters

n

goto next searched word

N

goto previous searched word

SAVE/QUIT/LOAD

:W save current file

:q quit current file after saving

:q! quit current file without saving

VISUAL MODE

V(small)

Character wise selection

V (capital)

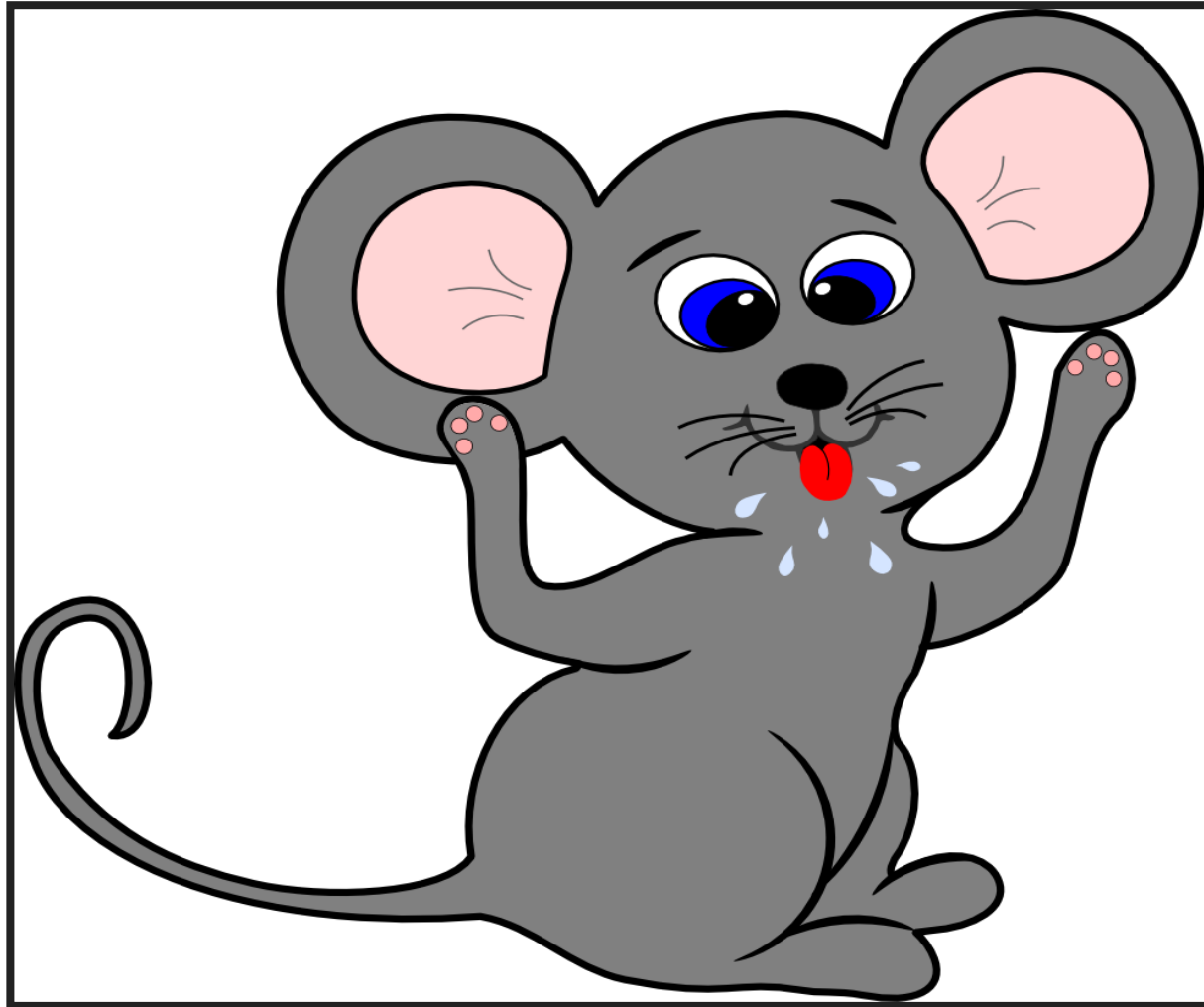
Block wise selection

SO ???

INVEST IN YOURSELF

DON'T START AT WORK

BREAK UP



STEAL YOUR FRIENDS' DOTFILES

KEEP PRACTICING

vimtutor

<http://www.openvim.com/>

LEARN. SHARE. SMILE

CARPE DIEM

make it your own