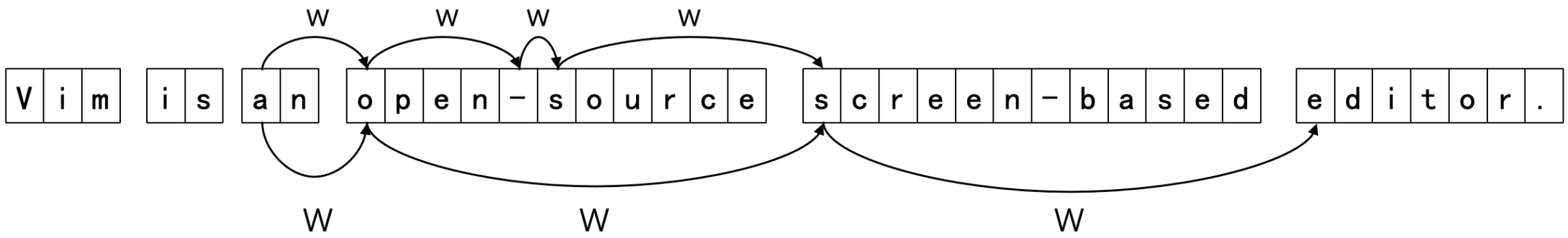
Vim：移动与编辑

基于单词的移动

简单的 `hjkl` 显然无法满足我们的需要

- `w`：代表“word”，跳转到下一处单词的开头
- `b`：代表“back”，跳转到上一处单词的开头
- `e`：代表“end”，跳转到下一处单词的结尾
- `ge`：`e` 的反向版本，跳转到上一处单词的结尾

`wbe` 大写版本 `WBE` 对应的单词是连续的非空字符



基于搜索的移动

行内搜索：

- `f{char}` / `t{char}`：跳转到本行下一个 `char` 字符出现处/出现前
- `;` / `,`：快速向后/向前重复 `ft` 查找
- `F{char}` / `T{char}`：往前搜索而非往后

文件中搜索：

- `/{pattern}`：跳转到本文件中下一个 `pattern` 出现的地方
- `?{pattern}`：跳转到本文件中上一个 `pattern` 出现的地方
- `pattern` 可以是正则表达式
- `*`：等价于 `/{pattern}`，`pattern` 是当前光标下的单词
- `nN`：快速重复 `/` 查找

基于标记的移动

- `m{mark}`：把当前位置标记为 `mark`
- `` `{mark} ``：跳转到名为 `mark` 的标记位置

`mark` 是 `a-z` 的字符

常用场景：当需要临时离开当前光标处，做一些事情后再**快速地**回来

我比较习惯用的标记是 `mm`

内置标记：

- ``` `` ```：上次跳转前的位置
- `` `. ``：上次修改的位置
- `` `^ ``：上次插入的位置

其它实用的跳转

- `^` / `$`：跳转到本行的开始/结尾
- `%`：跳到匹配的配对符（括号等）处

Operator+Motion=Action

`{operator}{motion}`：一次编辑动作

常见操作符：

- `c`：代表“change”，修改，删除内容并进入插入模式
- `d`：代表“delete”，删除
- `y`：代表“yank”，复制
- `v`：代表“visual”，选中文本，进入可视模式

例子：

- `dgg`：删除到第一行
- `ye`：复制到单词结尾
- `d$`：删除到行尾
- `dt;`：删除直到分号为止的内容

“操作符+移动” 是非常重要的操作逻辑，他允许你组合出各种动作

大部分操作符连续按两次（`cc/dd/yy`）：将其作用在这一行上

- `dd`：删除这一行

重复操作：`.` 命令

- `.`：重复上一次修改
- `u`：撤销上一次修改
- `<Ctrl-r>`：重做上一次修改

`.` 命令非常适合用于**需要多次重复某一个修改动作的场景**

省去了重复输入命令，大大提高效率

批量操作：数字+动作

`{count}{action}`：重复 `count` 次 `action` 动作

动作可以是移动动作或是编辑动作

- `4j`：向下移动 4 行
- `3dw`：删除 3 个单词
- `2yy`：复制 2 行
- `4p`：粘贴 4 次

数字+动作，是一种重要的批量操作方式，命令直观，语义明确

- `.` 命令可以直观地看到每一次的变化，在合适的时候停止
- 数字+动作则需要预先知道动作的次数

技巧：使用相对行号确定移动范围

当涉及行操作时，使用相对行号能够更直观地确定范围

`:set relativenumber`：开启

`:set norelativenumber`：关闭

小结

- 掌握进阶的移动与编辑命令
- （**重要**）掌握 `{operator}{motion}` 的操作哲学
- 掌握用 `.` 命令重复上一次修改
- （**重要**）掌握使用 `{count}{action}` 进行批量操作