链滴

使用 Vim 寄存器

作者：zwxbest

原文链接：https://ld246.com/article/1542942091793

来源网站：链滴

常见文本编辑器都会提供剪切板来支持复制粘贴，Vim 也不例外。 不同的是 Vim 提供了 10 类共 48 个寄存器，提供无与伦比的寄存功能。 最常用的 <code>y</code> 操作将会拷贝到默认的匿名存器中，我们也可以指定具体拷贝到哪个寄存器中。
一般来讲，可以用 <code>"{register}y</code> 来拷贝到 <code>{register}</code> 中， 用 code>"{register}p</code> 来粘贴 <code>{register}</code> 中的内容。例如： <code>"ayy</ ode> 可以拷贝当前行到寄存器 <code>a</code> 中，而 <code>"ap</code> 则可以粘贴寄存器 <code>a</code> 中的内容。
除了 <code>a-z</code> 26 个命名寄存器，Vim 还提供了很多特殊寄存器。合理地使用可以大地提高效率。例如：

<code>"+p</code> 可以粘贴剪切板的内容，
<code>":p</code> 可以粘贴上一个 Vim 命令（比如你刚刚费力拼写的正则表达式），
<code>"/p</code> 可以粘贴上一次搜索关键词（你猜的没错，正是 normal 模式下的 <code /foo</code> 搜索命令）。

在 Vim 中可通过 <code>:reg</code> 来查看每个寄存器当前的值。
<h2 id="寄存器分类">寄存器分类</h2>
Vim 提供了 10 类寄存器，可在 Vim 中通过 <code>:help registers</code> 查看帮助。

匿名寄存器 <code>""</code>
编号寄存器 <code>"0</code> 到 <code>"9</code>
小删除寄存器 <code>"-</code>
26 个命名寄存器 <code>"a</code> 到 <code>"z</code>
3 个只读寄存器 <code>":</code>, <code>".</code>, <code>"%</code>
Buffer 交替文件寄存器 <code>"#</code>
表达式寄存器 <code>"=</code>
选区和拖放寄存器 <code>"*</code>, <code>"+</code>, <code>"~</code>
黑洞寄存器 <code>"_</code>
搜索模式寄存器 <code>"/</code>

<h2 id="1-匿名寄存器">1. 匿名寄存器</h2>
使用 <code>d</code>, <code>c</code>, <code>s</code>, <code>x</code> 等会删字符的命令时，被删除字符会进入匿名寄存器 <code>""</code>。 你可以认为 <code>""</code> 寄存器是一个指针，指向刚才被存到的寄存器。
在如何用 Vim 搭建 IDE？一中提到，Mac 下可通过下列设置来让 Vim 共享系统剪切板， 就是这个原理：所有删除和拷贝操作默都会到匿名寄存器。
<pre><code class="highlight-chroma">set clipboard=unnamed

</code></pre>
使用 <code>y</code> 命令未指定寄存器会存到 <code>"0</code> 寄存器中，同时 <code ""</code> 会与该寄存器保有同样的值。 这意味着你使用 <code>p</code> 和 <code>"p</cod > 总会得到同样的结果。
<h2 id="2-编号寄存器">2. 编号寄存器</h2>
编号寄存器从 <code>"0</code> 到 <code>"9</code> 共 10 个，其中 <code>"0</code> 保存着拷贝来的字符串，<code>"1</code> 到 <code>"9</code> 保存着删除掉的字符串。 删操作符包括 <code>s</code>, <code>c</code>, <code>d</code>, <code>x</code>。 删掉的字符串会被存到 <code>"1</code> 中，上次删除的则会被存到 <code>"2</code> 中。以类推，Vim 会保存你最近的 9 次删除。

只有整行整行的删除，和通过段落级别的移动指令（包括 <code>%,(,),/,</code>,?,n,N,{,}<cod >） 的删除才会被放到</code>"1` 中。

当用户指定拷贝操作的寄存器时（如 <code>"ap</code>），<code>"0</code> 不会被写；但删除操作一定会被写入到 <code>"1</code> 中。

<blockquote>
<code>"0</code> 寄存器很有用，比如我们 copy 了一段文本然后用它替换另一段文本。 这默认寄存器 <code>""</code> 中的值就变成了被替换文本，如果还需要用 copy 的文本继续替换话就需要 <code>"0p</code> 了。
</blockquote>
<h2 id="3-小删除寄存器">3. 小删除寄存器</h2>
不足一行的小删除则会被放到小删除寄存器中（<code>"-</code>），起作用的删除操作符也括 <code>s</code>, <code>c</code>, <code>d</code>, <code>x</code>。 例如：
<pre><code class="highlight-chroma">dw   # 删除一个词
d9l   # 删除9个字符
cb   # 向前更改一词

</code></pre>
与 <code>"0</code> 寄存器类似，当用户指定寄存器并进行删除时，<code>"-</code> 不被写入。
<h2 id="4-命名寄存器">4. 命名寄存器</h2>
命名寄存器有 <code>"a</code> 到 <code>"z</code> 共 26 个，这些寄存器只有当我们指时才会被使用。 其实我们在录制宏时，所有键盘操作会以字符串的形式存到寄存器中。 例如录制一宏存到 <code>"a</code> 寄存器中，内容为更改当前行 <code>cc</code>，改为 <code>foo<code> 字符串：
<pre><code class="highlight-chroma">qaccfoo

</code></pre>
然后执行 <code>:reg</code> 来查看寄存器，可以发现 <code>a</code> 寄存器的值是 <cde>ccfoo</code>。
<blockquote>
小技巧：当使用小写字母进行操作时会覆盖当前寄存器内容，当使用大写字母进行操作时，会追当前寄存器内容。
</blockquote>
<h2 id="5-只读寄存器">5. 只读寄存器</h2>
只读寄存器共 3 个，它们的值是由 Vim 提供的，不允许改变：

<code>".</code>：上次 insert 模式中插入的字符串。还记得吗？<code>.</code> 命令可重复上次操作，而 <code>".</code> 存储了上次插入。
<code>"%</code>：当前文件名，不是全路径，也不是纯文件名，而是从当前 Vim 的工作目到该文件的路径。例如此时 Harttle 的 Vim 中，<code>"%p</code> 的结果为 <code>_drafts/vi-registers.md</code>。
<code>":</code>：上次命令模式下键入的命令。正如 <code>@a</code> 可以执行 <code"a</code> 寄存器中的宏一样，<code>@:</code> 可以执行上次命令。

<h2 id="6-交替文件寄存器">6. 交替文件寄存器</h2>
<p>交替文件寄存器 <code>"#</code> 存储着当前 Vim 窗口（Window）的交替文件。<strong交替文件（alternate file）是指 Buffer 中的上一个文件，可通过 <code>Ctrl+^</code> 来切换交替文件与当前文件。<p>
<blockquote>
Window 和 Buffer 有什么区别？参见 <a href="https://ld246.com/forward?goto=https%3

%2F%2Fharttle.land%2F2015%2F11%2F14%2Fvim-window.html" target="_blank" rel="nofollow ugc">Vim 多文件编辑：窗口一文。
</blockquote>
<h2 id="7-表达式寄存器">7. 表达式寄存器</h2>
表达式寄存器 <code>"=</code> 主要用于计算 Vim 脚本的返回值，并插入到文本中。 当我键入 <code>"=</code> 后光标会移动到命令行，此时我们可以输入任何 Vim 脚本的表达式。 例如 <code>3+2</code>，按下回车并且 <code>p</code> 则会得到 <code>5</code>。
这在我们调试 Vim 脚本时非常有用，比如调用一个函数看它是否有正确的返回值。
<h2 id="8-选择和拖放寄存器">8. 选择和拖放寄存器</h2>
选择和拖放寄存器包括 <code>"*</code>, <code>"+</code>, 和 <code>"~</code>，这个寄存器的行为是和 GUI 相关的。
<code>"*</code> 和 <code>"+</code> 在 Mac 和 Windows 中，都是指系统剪切板（clipboard），例如 <code>"*yy</code> 即可复制当前行到剪切板。 以供其他程序中粘贴。其他程序中制的内容也会被存储到这两个寄存器中。 在 X11 系统中（绝大多数带有桌面环境的 Linux 发行版）二者是有区别的：

<code>"*</code> 指 X11 中的 PRIMARY 选区，即鼠标选中区域。在桌面系统中可按鼠标中粘贴。
<code>"+</code> 指 X11 中的 CLIPBOARD 选区，即系统剪切板。在桌面系统中可按 Ctrl+V 粘贴。

<blockquote>
上文所述的 Mac 下 <code>set clipboard=unnamed</code> 会使得系统剪切板寄存器 <code>"*</code> 和 Vim 默认的匿名寄存器 <code>""</code> 始终保有同样的值，即 Vim 和系统共剪切板。
</blockquote>
有文本拖拽到 Vim 时，被拖拽的文本被存储在 <code>"~</code> 中。Vim 默认的行为是将 <code>"~</code> 中内容插入到光标所在位置。 当然你可以给 `` 做键盘映射。
<h2 id="9-黑洞寄存器">9. 黑洞寄存器</h2>
黑洞寄存器 <code>"_</code>，所有删除或拷贝到黑洞寄存器的文本将会消失。 这是为了在除文本的同时不影响任何寄存器的值，<code>"_</code> 通常用于 Vim 脚本中。
<h2 id="10-搜索寄存器">10. 搜索寄存器</h2>
搜索寄存器 <code>"/</code> 用于存储上一次搜索的关键词。Vim 中如何进行搜索呢？ 在 normal 模式下按下 <code>/</code> 即进入 search 模式，输入关键字并按下回车即可。
该寄存器是可写的，例如 <code>:let @/ = "harttle"</code> 将会把 <code>"harttle"</code> 写入该寄存器。 下次搜索时不输入搜索词直接回车便会搜索 <code>"harttle"</code>。
<h2 id="命令行模式拷贝">命令行模式拷贝</h2>
值得一提的时，任何寄存器中的值都是可以拷贝到命令模式下的。
比如对于寄存器 <code>"a</code> 中的值，在 normal 模式下可以通过 <code>"ap</code> 来粘贴；在 command-line 模式下通过 <code>a</code> 来粘贴。这一操作存在风险，因为寄存中的值可能是从网页中拷贝来的。
如果寄存器中的字符串存在 <code>字符或</code> 字符，则会时 Vim 回到 normal 模式，继续执行寄存器中的命令。为了防范_剪切板劫持_，可以添加下列的 Vim 配置：
<pre><code class="highlight-chroma">inoremap <C-r>+ <C-g>u<C-\><C-o>"+gP

</code></pre>
<blockquote>
该命令的解释请移步：http://vim.wikia.com/wiki/Pasting_registers
</blockquote>