Evitar asignaciones mentales

Los lectores no tienen que traducir mentalmente sus nombres en otros que ya conocen. Este problema suele aparecer al elegir entre no usar términos de dominio de problemas o de soluciones.

Es un problema de los nombres de variables de una sola letra. Un contador de bucles se podría bautizar como `i`, `j` o `k` (pero nunca `l`) si su ámbito es muy reducido y no hay conflictos con otros nombres, ya que los nombres de una letra son tradicionales en contadores de bucles. Sin embargo, en otros contextos, un nombre de una letra es una opción muy pobre: es como un marcador de posición que el lector debe asignar mentalmente a un concepto real. No hay peor motivo para usar el nombre `c` que `a` y `b` ya estén seleccionados.

Por lo general, los programadores son gente inteligente. A la gente inteligente le gusta presumir de sus habilidades mentales. Si puede recordar que `r` es la versión en minúscula de una URL sin el host y el sistema, debe ser muy listo.

Una diferencia entre un programador inteligente y un programador profesional es que este último sabe que la *claridad es lo que importa*. Los profesionales usan sus poderes para hacer el bien y crean código que otros puedan entender.

Nombres de clases

Las clases y los objetos deben tener nombres o frases de nombre como `Customer`, `WikiPage`, `Account` y `AddressParser`. Evite palabras como `Manager`, `Processor`, `Data`, o `Info` en el nombre de una clase. El nombre de una clase no debe ser un verbo.

Nombres de métodos

Los métodos deben tener nombres de verbo como `postPayment`, `deletePage` o `save`. Los métodos de acceso, de modificación y los predicados deben tener como nombre su valor y usar como prefijo `get`, `set` e `is` de acuerdo al estándar de javabean[9].

```
string name = employee.getName();
customer.setName("mike");
if (paycheck.isPosted())…
```

Al sobrecargar constructores, use métodos de factoría estáticos con nombres que describan los argumentos. Por ejemplo:

```
Complex fulcrumPoint = Complex.FromRealNumber(23.0);
```

es mejor que:

```
Complex fulcrumPoint = new Complex(23.0);
```

Refuerce su uso convirtiendo en privados sus constructores correspondientes.

No se exceda con el atractivo

Si los nombres son demasiado inteligentes, sólo los recordarán los que compartan el sentido del humor de su autor, y sólo mientras se acuerden del chiste. ¿Sabrán qué significa la función `HolyHandGrenade`? Sin duda es atractiva, pero en este caso puede que `DeleteItems` fuera más indicado. Opte por la claridad antes que por el entretenimiento. En el código, el atractivo suele aparecer como formas coloquiales o jergas. Por ejemplo, no use `whack()` en lugar de `kill()`. No recurra a bromas culturales como `eatMyShorts()` si quiere decir `abort()`.

Diga lo que piense. Piense lo que diga.

Una palabra por concepto

Elija una palabra por cada concepto abstracto y manténgala. Por ejemplo,