Enseñando C++

Adrian Kosmaczewski

2018-07-17

Una vez se me ocurrió enseñarle C++ a un amigo. Un error garrafal, pero bueno, fundamentalmente, lo que le dije fue esto.

1. La memoria asignada a un proceso está dividida en pila ("stack"), montón ("heap") y "static store". Esto es así desde que Von Neumann inventó la computadora.
2. Cada vez que ves un par de `{ }` en el código, es un nuevo "stack frame" o "contexto" que se crea en la pila. Todas las variables locales van ahí. Todas.
3. A diferencia de muchos otros lenguajes, C++ permite crear objetos en la pila:

```
string s { "test" };
vector<int> v {1, 2, 3};
```

Estas dos instrucciones crean objetos en la pila. Estos objetos **desaparecen** cuando llega el `}` de cierre del contexto o stack frame actual.

4. Pero también se pueden crear objetos en el montón, y para que no se pierdan, les asignamos un "puntero" en la pila:

```
string *s = new string { "test" };
vector<int> *v = new vector<int>  { 1, 2, 3 };
```

El string y el vector están en el montón; en la pila tenemos dos punteros (que básicamente contienen la dirección en memoria de dónde están los objetos en si).

5. Si pierdes los punteros, tenés un "memory leak" ya que **es imposible** recuperar la posición original de los objetos en el montón.
6. Si usas la palabra clave `static` tu objeto termina en el… static store.
7. Obviamente los corchetes de inicialización de los objetos no son los mismos corchetes que los que se usan para delimitar stack frames.
8. Lo importante es que asimiles las dos sintaxis de creación ya que tienen semánticas diferentes.
9. El montón se usa para objetos que viven larga vida o que son enormes; el stack es perfecto para objetos pequeños, temporarios y para valores escalares como enteros o números de coma flotante.

10. Cada vez que usas “new” sos el dueño del objeto y por lo tanto es obligatorio que uses “delete” cuando no necesitas más el objeto.
11. Java, PHP, C# usan un “recolector de basura” que escanea cada tanto el montón y borra los objetos que ya no tienen dueño. C++ no tiene eso.
12. C++ tiene una librería estándar impresionante pero por razones históricas y/o técnicas muchas librerías vienen con sus propios string, array, etc, y hay que aprender a usarlos, a sabiendas de que en otro proyecto, quizás, uses otro Array u otro string. Es lo que hay.

Mas cosas interesantes aca: http://cppquiz.org