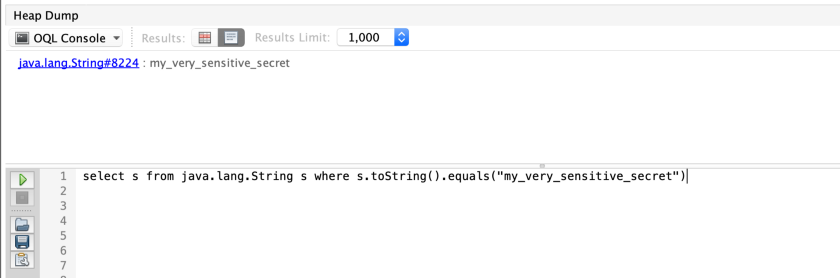
The Problem With Storing Passwords in String

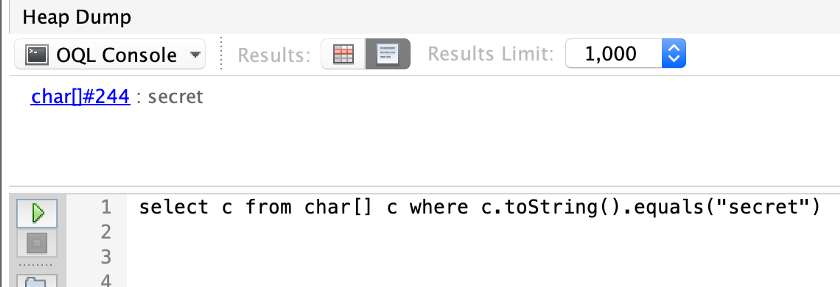


Strings in Java are immutable, which means that once created they never change. Furthermore since they are a relatively expensive resource, they are pooled in the internal String pool. This means that it will take a lot of time for them to be dereferenced and then collected by the GC.

Here we have two passwords, one is stored in string and other in char[]. I will be using [Java VisualVM](https://visualvm.github.io/) to illustrate how we can easily find the ‘my\_very\_sensitive\_secret’ in the HeapDump by using OQL (Object Query Language)



If we try the same for the second variable ‘secret’ we will see that it is just as easy to get it



So what is the point? Why would I ever bother with the array, which is harder to use. Actually the only benefit the array provides in this case is that it is not immutable. This allows us to get rid of the password the moment we no longer use it.

The string pool will reference the string for a long period of time whereas the char array is eligible for collection after line 4. Essentially we are reducing the time the password stays in the heap.

As a conclusion I should mention that keeping the password in plain text is not a good idea and should be avoided if possible. If we keep the password encrypted or hashed we won’t have to worry about string pools and GC cycles, because even if someone manages to dump the heap the string will be unreadable.