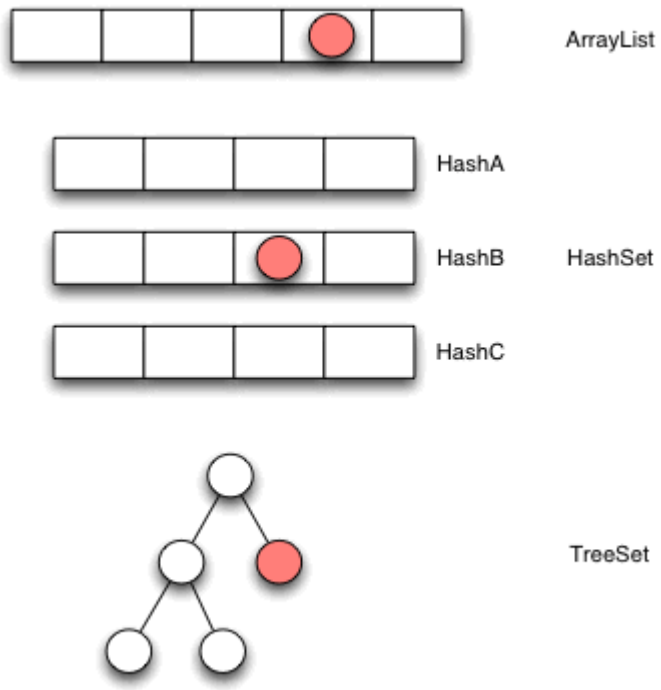
Un tema muy recurrente cuando hablamos de rendimiento es el de Java Collections Performance. Todos usamos el framework de collecciones pero muchas veces nos olvidamos de su rendimiento. ¿Cual es la colección más rápida a la hora de buscar elementos?. Vamos a verlo con un ejemplo sencillo. Para ello partiremos de tres colecciones diferentes, un ArrayList,un HashSet y un TreeSet. Todos ellos soportan el interface Collection y podemos buscar un elemento a través del método contains().



Hay que recordar que un ArrayList almacena la información en formato lista. Un HashSet lo almacena como un bloque de claves hash y cada una dispone de su propia sublista. Por último un TreeSet almacena un conjunto en forma de árbol binario. Vamos a usar la clase System y su método nanoTime para obtener los tiempos a detalle:

```
package com.arquitecturajava;
```

```
import java.util.ArrayList;
import java.util.HashSet;
import java.util.List;
import java.util.Set;
import java.util.TreeSet;
import java.util.concurrent.TimeUnit;

public class Principal {

 public static void main(String[] args) {

 List<String> lista = new ArrayList<String>();

 for (int i = 0; i < 1000000; i++) {

 lista.add("item" + i);

 }

 long tiempo1 = System.nanoTime();

 System.out.println(lista.contains("item4999"));

 long tiempo2 = System.nanoTime();

 System.out.println(TimeUnit.NANOSECONDS.toMicros(tiempo2 - tiempo1));

 Set<String> conjunto = new HashSet<String>();

 for (int i = 0; i < 1000000; i++) {
```

```
conjunto.add("item" + i);

}



long tiempo3 = System.nanoTime();

System.out.println(conjunto.contains("item4999"));

long tiempo4 = System.nanoTime();


System.out.println(TimeUnit.NANOSECONDS.toMicros(tiempo4 - tiempo3));

Set<String> conjunto2 = new TreeSet<String>();

for (int i = 0; i < 1000000; i++) {

conjunto2.add("item" + i);

}

long tiempo5 = System.nanoTime();

System.out.println(conjunto2.contains("item4999"));

long tiempo6 = System.nanoTime();
```

```
 System.out.println(TimeUnit.NANOSECONDS.toMicros(tiempo6 - tiempo5));

 }

}
```

El método System.nanoTime nos devuelve la información en nanosegundos , 10 a la -9 segundos. Utilizamos la enumeración TimeUnit y lo convertimos a microsegundos una medída más habitual.

```
true
780
true
131
true
115
```

La diferencia entre un HashSet y un ArrayList es grande ya que un hash nos permite buscar en un subconjunto de los datos. Por último el rendimiento de un TreeSet supera al de un HashSet al estar ordenado.

Otros artículos relacionados: List vs Set , Java Generics , Java Fluid Interface