Colecciones paramétricas

DigitalHouse >
Coding School

Certified Tech Developer
The Ultimate Degree

Índice

1 Introducción

Introducción

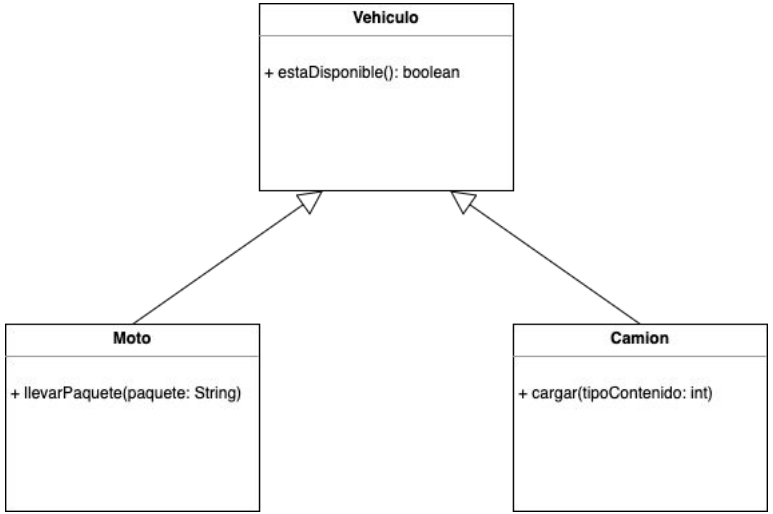
Conoceremos, en esta presentación, el uso de la programación paramétrica en colecciones. Para comenzar, recordemos que en todas las operaciones que podemos hacer sobre las colecciones, el tipo utilizado es **Object**.

- add(**Object** o) : void
- get(int i) : **Object**
- iterator() : iterator
    - hasNext() : boolean

next() : **Object**

Como en Java, todas nuestras clases heredan de Object, entonces, podemos mezclar objetos de diferentes tipos en la misma colección.

Para analizar en detalle el problema, asumamos que tenemos una serie de vehículos donde nuestra empresa debe administrar lo que transportan.



Podríamos hacer una lista con los vehículos de nuestra empresa.

```
List vehiculos = new ArrayList();

Moto moto = new Moto();
Camion camion = new Camion();

vehiculos.add(moto);
vehiculos.add(camion);
```

Si en algún momento quisiéramos obtener algún vehículo de la lista, debemos castear.

```
Moto moto = (Moto) vehiculos.get(0);
Camion camion = (Camion) vehiculos.get(1);
```

casting

Si quisiéramos recorrer la lista de vehículos y ver si están disponibles, podríamos hacer esto.

```
for(Object o :vehiculos) {
   System.out.println(((Vehiculo)o).estaDisponible());
}
```

Pero, si quisiéramos agregar carga al camión, es necesario castear solo a aquellos elementos de la lista que sean camiones. De otra manera, **tendríamos un error de ejecución.**

```
for(Object o :vehiculos) {
    if( o instanceof Camion)
        ((Camion)o).cargar(“papas”);
}
```

2 Solución con Generics

Solución con Generics

Para **evitar mezclar objetos de diferente tipo en una colección**, a partir de la versión 1.5 de Java todas las colecciones pueden recibir como parámetro el tipo, es decir, soportan Generics.

```
List<Camion> vehiculos = new ArrayList<Camion>();
```

Si usamos las colecciones parametrizando su tipo, tendremos un control en tiempo de compilación sobre los tipos de los objetos que agregamos a la colección, de forma tal que, al momento de correr la colección, no hace falta chequear por el instanceof dado que no deberemos castear, porque no podríamos “mezclar” tipos de objetos.

```
for(Camion o :vehiculos) {
        o.cargar(“papas”);
}
```


DigitalHouse>
Coding School