

Prácticas POO

Curso 10/11

Alejandro Bellogín



Escuela Politécnica Superior
Universidad Autónoma de Madrid
Marzo 2011

<http://www.eps.uam.es/~abellogin>

Práctica 3

- Implementación sistema de gestión de tienda virtual
- Con interfaz gráfica
- Sincronizado
- Distribuido (RMI)

Práctica 3

- Implementación sistema de gestión de tienda virtual
- Con interfaz gráfica 
- Sincronizado
- Distribuido (RMI) 

Sincronización

- Asegurar consistencia en operaciones
- Comprobación:
 - Lanzar dos clientes y hacer la misma operación a la vez
- Alternativas
 - Semáforos (*java.util.concurrent.Semaphore*)
 - Hacer *synchronized* una variable, un método o un bloque de código

Comprobación de sincronización

```
final Tienda tienda = new Libreria();  
final Artículo a = tienda.getArticulos().get(0);  
final Cliente c1 = tienda.getCliente("1");  
final Cliente c2 = tienda.getCliente("2");  
Thread hilo1 = new Thread() {
```

```
    @Override  
    public void run() {  
        c1.meterAlCarrito(a, 1);  
        tienda.venderCarrito(c1);  
    }  
};
```

```
Thread hilo2 = new Thread() {
```

```
    @Override  
    public void run() {  
        c2.meterAlCarrito(a, 1);  
        tienda.venderCarrito(c2);  
    }  
};
```

```
hilo2.start();  
hilo1.start();  
Thread.sleep(1000);  
tienda.mostrarStock(a);
```

```
public void venderCarrito(Cliente cl) {  
    for (Artículo a : cl.getCarrito()) {  
        almacen.put(a, almacen.get(a) - 1);  
    }  
    cl.getCarrito().clear();  
}
```



Alternativas para sincronizar

```
public synchronized void venderCarrito(Cliente cl) {
    for (Articulo a : cl.getCarrito()) {
        almacen.put(a, almacen.get(a) - 1);
    }
    cl.getCarrito().clear();
}

public void venderCarrito(Cliente cl) {
    synchronized (almacen){
        for (Articulo a : cl.getCarrito()) {
            almacen.put(a, almacen.get(a) - 1);
        }
    }
    cl.getCarrito().clear();
}

public Libreria() {
    almacen = new ConcurrentHashMap<Articulo, Integer>();
}

public void venderCarrito(Cliente cl) {
    for (Articulo a : cl.getCarrito()) {
        almacen.put(a, almacen.get(a) - 1);
    }
    cl.getCarrito().clear();
}
```

```
private Semaphore sem;

public Libreria() {
    almacen = new HashMap<Articulo, Integer>();
    almacen.put(new Articulo("test"), 2);
    sem = new Semaphore(1);
}

public void venderCarrito(Cliente cl) {
    try {
        sem.acquire();
    } catch (InterruptedException e) {
    }
    for (Articulo a : cl.getCarrito()) {
        almacen.put(a, almacen.get(a) - 1);
    }
    sem.release();
    cl.getCarrito().clear();
}
```