

## Explicación - Ejercicio 05 - Estaciones eléctricas

### Introducción

En este ejercicio se pide hacer la gestión de estaciones eléctricas usando los Semáforos (`java.util.concurrent.Semaphore`).

Se quiere desarrollar un software para simular un punto de carga de vehículos eléctricos donde el número de cargadores es limitado. La intención es controlar la concurrencia y asegurar que sólo un número fijo de vehículos puedan cargar al mismo tiempo.

Un parking dispone de 5 estaciones de carga eléctrica:

- El tiempo de carga es variable
- Los vehículos llegan de forma asíncrona (como hilos) y deben esperar si no hay estaciones libres.

### EstacionDeCarga

Gestiona el acceso controlado a 5 puntos de carga usando semáforos

#### ¿Qué es un Semaphore y cómo funciona?

Un Semaphore es un mecanismo de control que gestiona un número limitado de permisos:

- **Crear semáforo con N permisos:** `new Semaphore(5)` -> Hay 5 permisos disponibles
- **acquire():** Un hilo solicita un permiso. Si hay disponibles, lo obtiene y continúa. Si no hay, el hilo se bloquea y espera hasta que se libere uno
- **release():** Un hilo devuelve el permiso, quedando disponible para otros hilos en espera

**Ejemplo real - Estacionamiento de hospital:** Un hospital tiene un estacionamiento con 5 plazas reservadas para ambulancias:

- **Semaphore(5):** Hay 5 plazas disponibles
- **Ambulancia llega -> acquire():** Ocupa una plaza (quedan 4 libres)
- **Ambulancia sale -> release():** Libera la plaza (vuelven a ser 5 disponibles)
- **Si las 5 están ocupadas:** La ambulancia 6 espera en la entrada hasta que salga otra

#### Atributos

- **Semaphore:** Controla que solo 5 vehículos acceden a la vez.
- **boolean[] puestosOcupados:** Array que indica qué puestos están ocupados (true) o libres (false)
- **Object bloqueoPuestos:** Objeto para sincronizar el acceso al array y evitar que dos hilos entren en el mismo puesto.

#### Métodos:

- **solicitarAcceso():** Pide permiso al semáforo. Si no hay permisos disponibles, el hilo espera.
- **liberarAcceso():** Devuelve el permiso al semáforo para que otro vehículo pueda entrar

- **asignarPuesto():** Busca el primer puesto libre en el array y lo marca como ocupado. Usa *synchronized* para evitar que dos hilos entren a la vez.
- **liberarPuesto():** Marca el puesto como libre. También sincronizado.

## VehiculoElectrico

Representa un vehículo que necesita cargar. Cada vehículo es una hilo independiente.

### Método run():

1. **Solicita acceso** -> Espera hasta que el semáforo dé permiso.
2. **Asigna puesto** -> Obtiene un número de puesto (1 - 5)
3. **Simula carga** -> Duerme entre 1 - 3 segundos (Random)
4. **Libera recursos** -> En el *finally* garantiza que siempre se libere el puesto y el permiso, incluso si hay errores.

El *finally* asegura que no se queden recursos bloqueados.

## Main

- Crea una estación con 5 puestos
- Lanza 10 vehículos
- Los primeros 5 entran inmediatamente
- Los otros 5 esperan a que se libere un puesto
- El sleep de 200ms hace que las llegadas sean escalonadas.