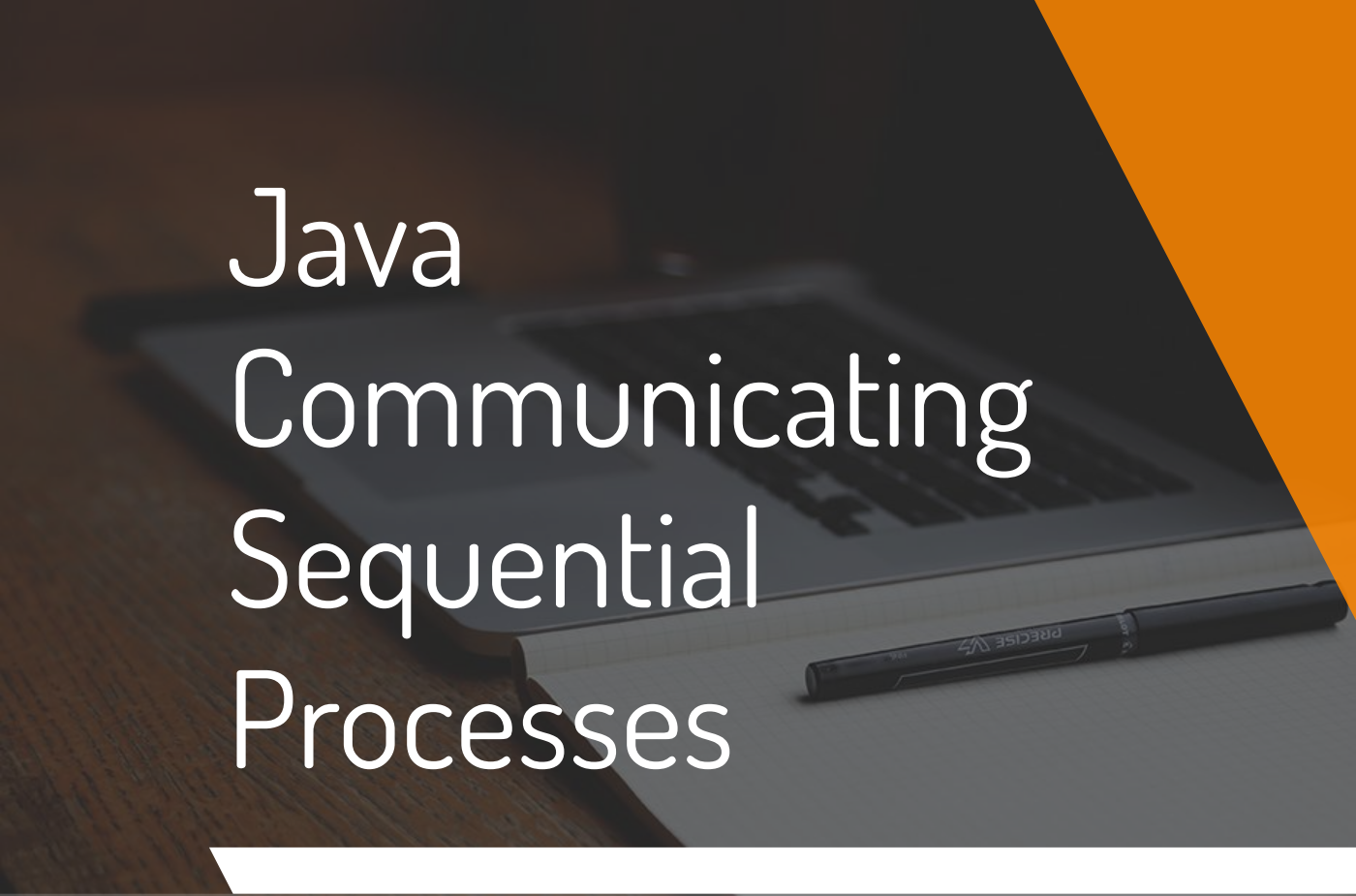
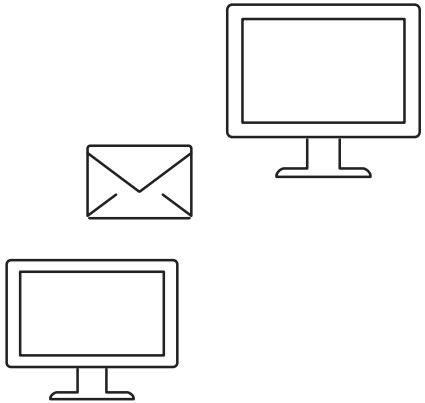
Java Communicating Sequential Processes



Martinez Buenrostro Jorge Rafael

1. Introducción

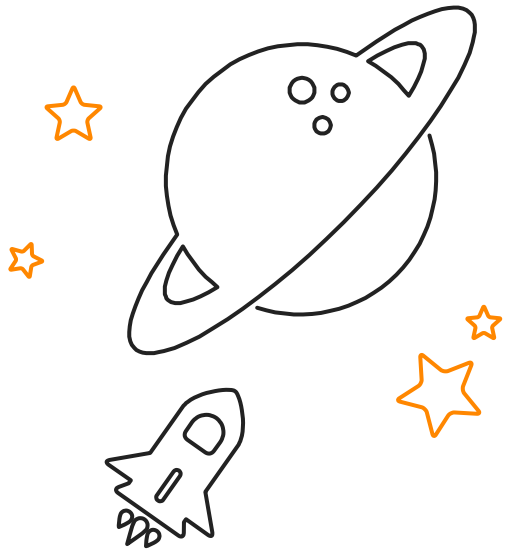
¿Qué es CSP? ¿Qué es Java? ¿Qué es JCSP?



¿Qué es CSP?

Communicating Sequential Processes

Es un modelo que se basa en el paso de mensajes sincrónicos a través de canales entre procesos secuenciales

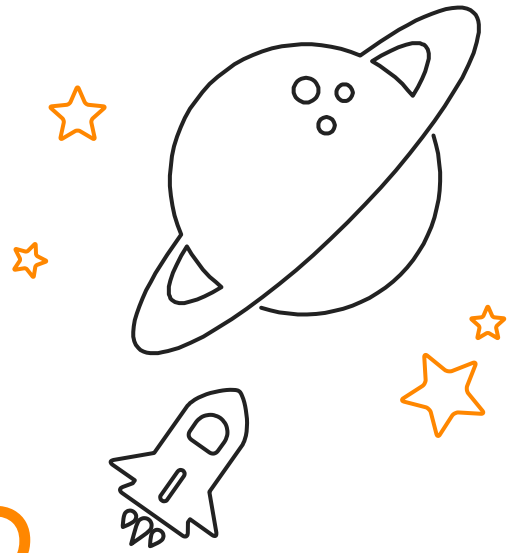


¿Qué es Java?

Es un lenguaje de programación orientado a objetos que ofrece soporte nativo para la programación concurrente mediante el uso de hilos

¿Qué es un hilo en Java?

- Unidad básica de ejecución
- Comparten memoria y recursos con otros hilos del mismo proceso
- La clase Thread encapsula el comportamiento y el estado del hilo



¿Qué es JCSP?

Es una implementación del modelo CSP para el lenguaje Java

Primitivas de sincronización

- Canales
- Temporizadores
- Equipos
- Barreras
- Cubetas

2.
JCSP

Entidades Activas, Comunicación, Sincronización

Entidades Activas

- Comunicación mediante canales
- Encapsulación del estado y comportamiento
- Procesamiento concurrente
- Receptividad selectiva
- Gestión propia de la concurrencia

Modelos de comunicación

- Síncrona
- Asíncrona
- Extremo a Extremo
- Selectiva
- Prioritaria

Sincronización

- Semáforos
- Barreras
- Alternativas
- Contadores de eventos
- Rendezvous

Antes del ejemplo

Canal

- Síncrono
- Unidireccional
- Tiene varios tipos:
  - One2One
  - Any2One
  - One2Any
  - Any2Any

Mensaje

- Cualquier tipo de dato
- Tamaño de 64KB
- Univaluado o multivaluado

Evento

- Los procesos se bloquean hasta que se produce el evento deseado

3. Ejemplo

```
public static void main (String [ ] args ){
    //Crear los canales de comunicacion
    Any2OneChannel<String> canal1=Channel.any2one();
    Any2OneChannel<String> canal2=Channel.any2one();

    // Crear los procesos
    CSProcess proceso1=new Proceso1 (canal1.out(),canal2.in());
    CSProcess proceso2=new Proceso2 (canal2.out(),canal1.in());

    //Ejecutar los procesos en paralelo
    Parallel parallel=new Parallel();
    parallel.addProcess (proceso1);
    parallel.addProcess (proceso2);

    //ejecuta los procesos concurrentes
    parallel.run();
}
```

```java
// Proceso 1
static class Proceso1 implements CSProcess {
    private ChannelOutput<String> salida;
    private ChannelInput<String> entrada;

    public Proceso1 (ChannelOutput<String> salida,ChannelInput<String> entrada){
        this.salida=salida;
        this.entrada=entrada ;
    }

    public void run(){
        //Enviar un mensaje al proceso 2
        salida.write("Hola desde Proceso 1") ;
        // Esperar la respuesta del proceso 2
        String respuesta=entrada.read();
        System.out.println("Proceso 1 recibi:"+respuesta) ;
    }
}
```

```
static class Proceso2 implements CSProcess {
    private ChannelOutput<String> salida;
    private ChannelInput<String> entrada;

    public Proceso2 (ChannelOutput<String> salida, ChannelInput<String> entrada){
        this.salida=salida;
        this.entrada=entrada ;
    }
    public void run (){
        // Esperar el mensaje del proceso 1
        String mensaje=entrada.read();
        System.out.println( "Proceso 2 recibi:" + mensaje);
        // Responder a l proceso 1
        salida.write("Hola desde Proceso 2");
    }
}
```

Resultado

```
Proceso 2 recibi: Hola desde Proceso 1
Proceso 1 recibi: Hola desde Proceso 2
```

Muchas gracias!

¿Alguna pregunta?