

Ayuda para Práctica 3

Sincronización con Semáforos

Descarga este documento en:

Enlace corto (puede no funcionar desde la facultad): <http://goo.gl/yQIJQ>

Enlace largo: <https://www.dropbox.com/s/q3ro833vkur4gds/Ayuda%20Practica%203.pdf>

Más ayuda: en tutorías o
Pablo.Bermejo@uclm.es

Recordad que...

- Hemos aprendido a **sincronizar** hilos a partir de una **variable condición** y:
 - **Busy wait** (espera activa) o
 - ***Wait*** y ***notify*** (señales nativas)
- Hemos aprendido a garantizar **exclusión mutua** con:
 - Algoritmo de Dekker
 - El modificador ***synchronized***

Ahora hay que...

- Garantizar exclusión mutua con... semáforos (ejercicio 1)
- Sincronizar con... semáforos también (ejercicio 2)

Ej 1.A

- 1) Ejecutad el Main3, no habrá problema porque solo se escriben 2 líneas y no se borra ninguna.
- 2) Siguiendo los parámetros del guion de prácticas:
 - Cambiad los métodos `operador1` y `operador2` para que escriban 20 veces en pantalla.
 - Cambiad el método `operador3` para que borre también 20 veces.
 - Recordad que tras cada acceso a pantalla hay que dormir usando *nap* (que duerman más tiempo al borrar que al escribir).

Ej 1.A

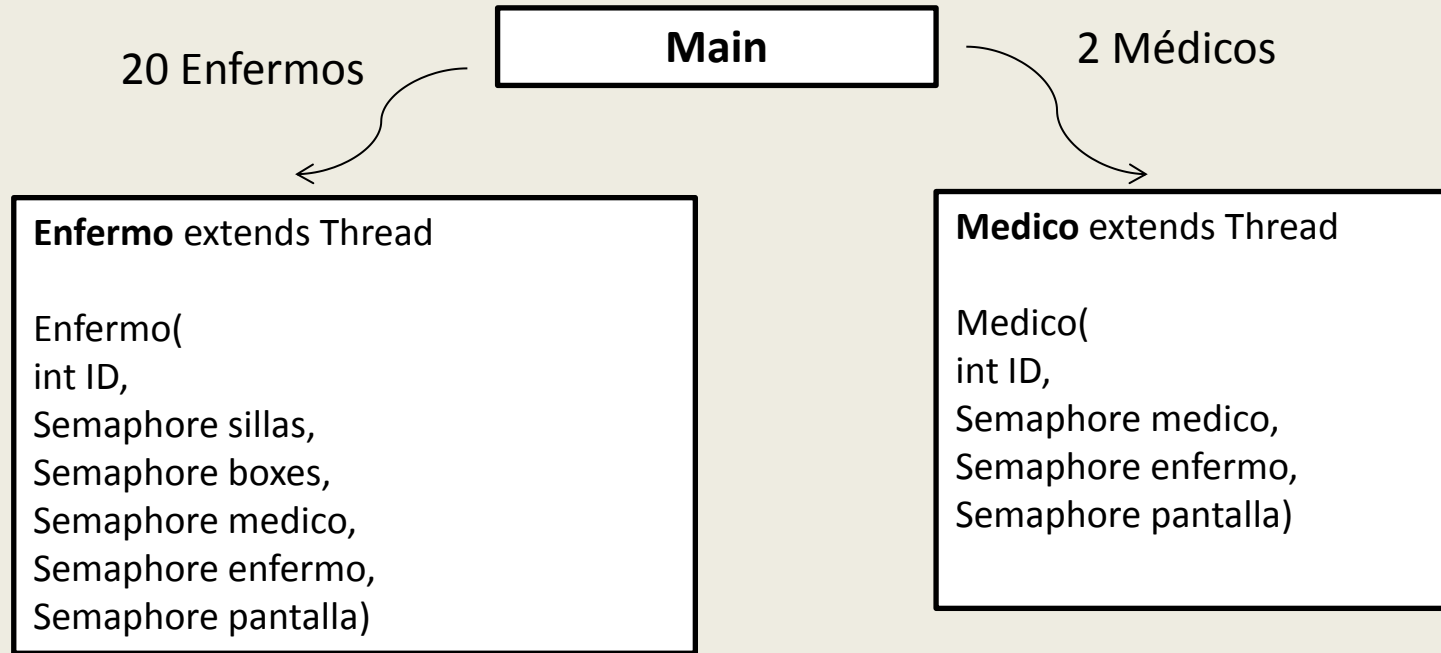
- 3) Hay que **garantizar exclusión mutua antes de acceder a pantalla**; en este caso, antes de acceder a `IPanel2.ponLinea` o `IPanel2.borraLinea`:
- Hay que utilizar **1** instancia del objeto `java.util.concurrent.Semaphore` que tendrá que ser compartida por todos los hilos: **¿variable global o local?**
 - Un semáforo reparte permisos para actuar; si no le quedan permisos para darle a un hilo, éste se esperará hasta que el semáforo se lo de.
 - Cuando el hilo acaba su trabajo, debe devolver el permiso al semáforo.
 - El semáforo mantiene un contador de permisos que dispone para dar actualmente; este contador nunca baja de 0.
 - Por ello, para garantizar exclusión mutua, **¿con qué valor hay que iniciar el semáforo?**

Ej 1.B

- La exclusión mutua, como ya sabemos, sirve para que haya código que solo se pueda ejecutar 1 vez.
- ¿Y si el código no está escrito de forma iterativa, sino repartido en varios objetos? → algoritmos como dekker no funcionan o se vuelven extremadamente complejos.
- Usa **synchronized** para garantizar que `IPanel2.ponLinea` y `IPanel2.borraLinea` se ejecutan solo 1 a la vez.

- Hemos visto en el ej. 1.A que un semáforo binario sirve para garantizar exclusión mutua
- Pero un semáforo sirve también para sincronizar... Ej. 2.A
- Antes de comenzar el ejercicio 2, conviene que le echéis un vistazo a la tabla Method Summary del javadoc de Semaphore:
<http://docs.oracle.com/javase/1.5.0/docs/api/java/util/concurrent/Semaphore.html>

Ej. 2.A



La clase Main será la encargada de crear los hilos de Enfermos y Médicos; y de instanciar los semáforos para pasarlos como parámetros:

- **Semáforos de recursos:** sillas, boxes y la pantalla
- **Semáforos de citas:** médicos y enfermos
- Los semáforos de recursos se inician al valor de la cantidad de recursos disponibles
- Los semáforos para sincronizar una cita se inician a 0

Ej. 2.A

- El semáforo *pantalla* se utilizará como exclusión mutua para escribir mensajes por pantalla. Debido a la alta cantidad de hilos, es posible que los mensajes se crucen o incluso sufran de inanición. En vez de `println`, usad el método *escribe*, en la clase `Medico` y la clase `Enfermo`.

```
protected void escribe(String mensaje) {
```

```
    try {  
        Spantalla.acquire();  
    } catch (InterruptedException e) {  
        e.printStackTrace();  
    }  
    System.out.println(mensaje);  
    Spantalla.release();  
}
```

Ej. 2.A

- Mira en el javadoc de Semaphore cómo instanciar el semáforo para asegurar *fairness*. Hacedlo para todos los semáforos.
- Es posible que tu programa esté bien, pero los mensajes por pantalla den la impresión de que está mal → cuando se libere un recurso y se escriba por pantalla la cantidad de recursos que quedan, los comandos correspondientes deben ir encerrados en un bloque *synchronized* o usando otro semáforo utilizado para exclusión mutua, llamado por ejemplo *mutex*, y ya no hace falta usar el metodo escribe.

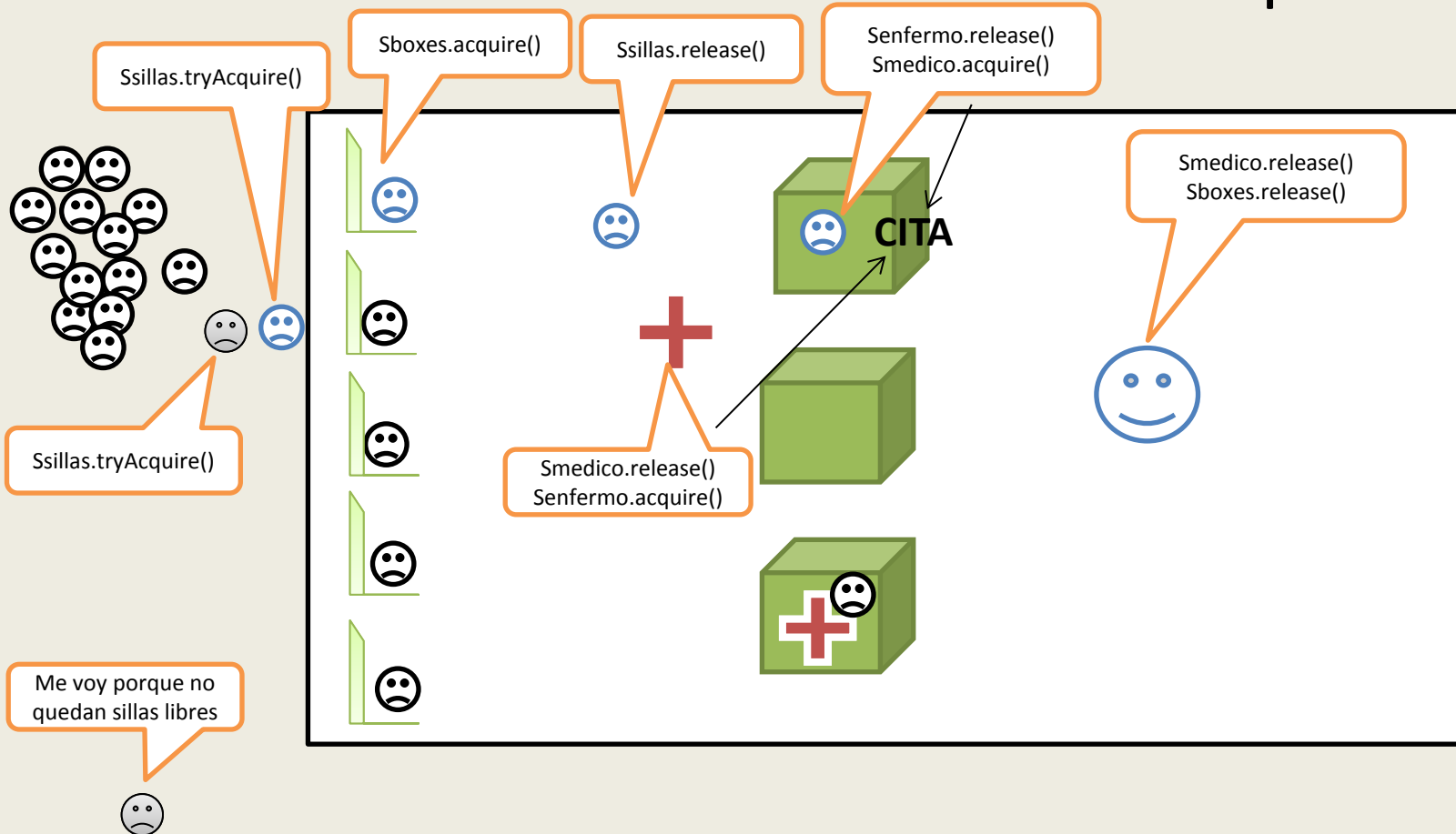
Ej. 2.A

- El hilo más simple es **Médico**, el cual solo espera a tener citas:

```
public void run() {  
  
    while (true) {  
        escribe("[Medico " + ID + "]: Esperando un paciente...");  
        // PROTOCOLO PARA CREAR UNA CITA  
        try {  
            ...// indicar que el médico está libre  
            ...// pedir un paciente  
            Thread.sleep((long)Math.random());  
        } catch (InterruptedException e) {  
            e.printStackTrace();  
        }  
  
    }  
  
}
```

Ej. 2.A

- El hilo Enfermo deberá hacer varios pasos



¡¡CUIDADO CON LOS MENSAJES POR PANTALLA!!

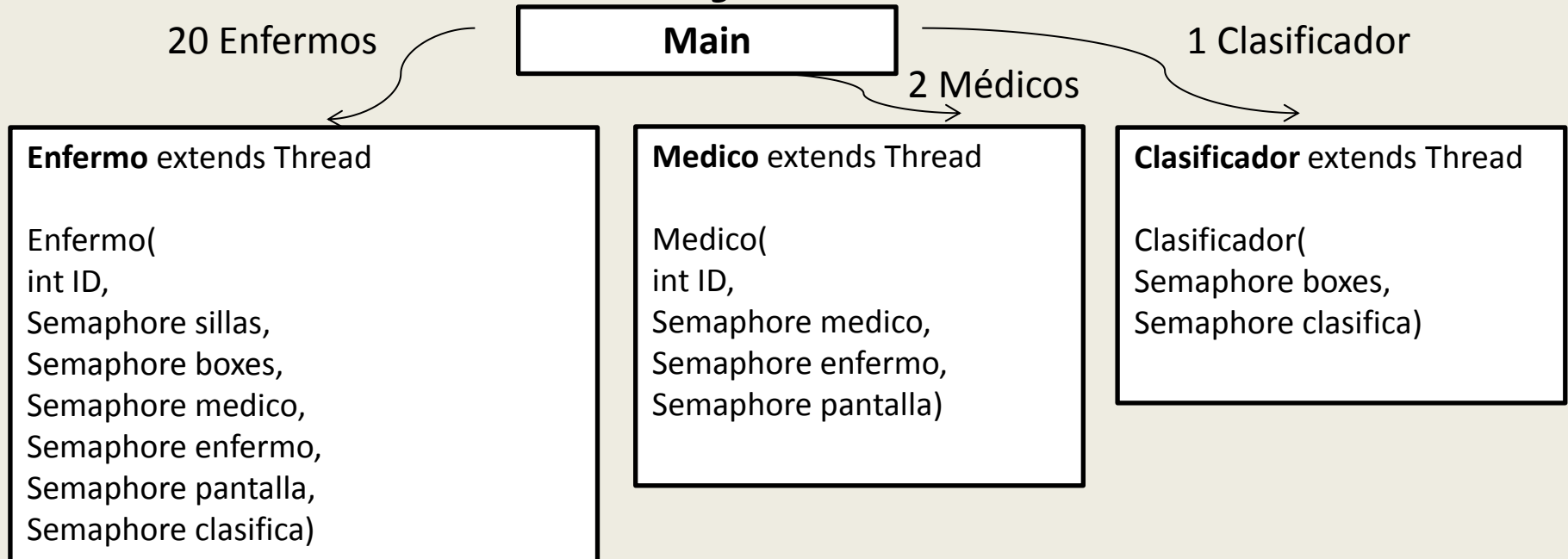
Ejemplo de salida correcta:

[Medico 1]: Esperando un paciente...
[Medico 2]: Esperando un paciente...
[1]: Me siento y quedan 4 sillas libres.
[1]: Entro en un box, dejo 5 sillas libres y espero un médico.
[3]: Me siento y quedan 4 sillas libres.
[4]: Me siento y quedan 3 sillas libres.
[2]: Me siento y quedan 2 sillas libres.
[5]: Me siento y quedan 1 sillas libres.
[6]: Me siento y quedan 0 sillas libres.
[1]: El médico me ha curado. ME VOY!!
[7]: Me voy porque no quedan sillas libres!
[3]: Entro en un box, dejo 1 sillas libres y espero un médico.
[10]: Me voy porque no quedan sillas libres!
[11]: Me voy porque no quedan sillas libres!
[14]: Me voy porque no quedan sillas libres!
[8]: Me siento y quedan 0 sillas libres.
[9]: Me voy porque no quedan sillas libres!
[4]: Entro en un box, dejo 1 sillas libres y espero un médico.
[12]: Me siento y quedan 0 sillas libres.
[16]: Me voy porque no quedan sillas libres!
[15]: Me voy porque no quedan sillas libres!
[Medico 1]: Esperando un paciente...
[20]: Me voy porque no quedan sillas libres!
[24]: Me voy porque no quedan sillas libres!
[18]: Me voy porque no quedan sillas libres!
[19]: Me voy porque no quedan sillas libres!
[21]: Me voy porque no quedan sillas libres!
[13]: Me voy porque no quedan sillas libres!
[28]: Me voy porque no quedan sillas libres!
[22]: Me voy porque no quedan sillas libres!
[23]: Me voy porque no quedan sillas libres!
[25]: Me voy porque no quedan sillas libres!
[17]: Me voy porque no quedan sillas libres!
[26]: Me voy porque no quedan sillas libres!

[29]: Me voy porque no quedan sillas libres!
[27]: Me voy porque no quedan sillas libres!
[2]: Entro en un box, dejo 1 sillas libres y espero un médico.
[30]: Me siento y quedan 0 sillas libres.
[31]: Me voy porque no quedan sillas libres!
[32]: Me voy porque no quedan sillas libres!
[33]: Me voy porque no quedan sillas libres!
[34]: Me voy porque no quedan sillas libres!
[3]: El médico me ha curado. ME VOY!!
[Medico 2]: Esperando un paciente...
[35]: Me voy porque no quedan sillas libres!
[37]: Me voy porque no quedan sillas libres!
[36]: Me voy porque no quedan sillas libres!
[38]: Me voy porque no quedan sillas libres!
[39]: Me voy porque no quedan sillas libres!
[40]: Me voy porque no quedan sillas libres!
[41]: Me voy porque no quedan sillas libres!
[42]: Me voy porque no quedan sillas libres!
[43]: Me voy porque no quedan sillas libres!
[44]: Me voy porque no quedan sillas libres!
[45]: Me voy porque no quedan sillas libres!
[46]: Me voy porque no quedan sillas libres!
[4]: El médico me ha curado. ME VOY!!
[48]: Me voy porque no quedan sillas libres!
[50]: Me voy porque no quedan sillas libres!
[49]: Me voy porque no quedan sillas libres!
[47]: Me voy porque no quedan sillas libres!
[Medico 1]: Esperando un paciente...
[2]: El médico me ha curado. ME VOY!!
[5]: Entro en un box, dejo 1 sillas libres y espero un médico.
[Medico 2]: Esperando un paciente...
[6]: Entro en un box, dejo 2 sillas libres y espero un médico.
[8]: Entro en un box, dejo 3 sillas libres y espero un médico.
[5]: El médico me ha curado. ME VOY!!

[Medico 1]: Esperando un paciente...
[6]: El médico me ha curado. ME VOY!!
[Medico 2]: Esperando un paciente...
[8]: El médico me ha curado. ME VOY!!
[12]: Entro en un box, dejo 4 sillas libres y espero un médico.
[Medico 1]: Esperando un paciente...
[30]: Entro en un box, dejo 5 sillas libres y espero un médico.
[12]: El médico me ha curado. ME VOY!!
[Medico 2]: Esperando un paciente...
[30]: El médico me ha curado. ME VOY!!
[Medico 1]: Esperando un paciente...

Ej 2.B



- Ahora es el Clasificador el que obtiene el box y se anuncia para obtener cita con el enfermo.
- El Enfermo pide cita con el Clasificador.

- Por favor, intentad sacar 5 minutos para contestar la siguiente encuesta anónima, me será de mucha utilidad. Gracias!

<http://goo.gl/g5iva>